BEST AVAILABLE IMAGES

Defective images within this document are accurate representations of the original documents submitted by the applicant.

Defects in the images may include (but are not limited to):

- BLACK BORDERS
- TEXT CUT OFF AT TOP, BOTTOM OR SIDES
- FADED TEXT
- ILLEGIBLE TEXT
- SKEWED/SLANTED IMAGES
- COLORED PHOTOS
- BLACK OR VERY BLACK AND WHITE DARK PHOTOS
- GRAY SCALE DOCUMENTS

IMAGES ARE BEST AVAILABLE COPY.

As rescanning documents *will not* correct images, please do not report the images to the Image Problem Mailbox.

A compact microelectronic device for performing modular multiplication and exponentiation over large numbers.

Patent number: EP0601907
Publication date: 1994-06-15
Inventor: GRESSEL CARMI DAVID (IL); DROR ITAI (IL); HADAD ISAAC (IL); AZARI BENJAMIN (IL); HENDEL DAVID (IL)
Applicant: FORTRESS U & T LTD (IL)
Classification:
- international: G06F7/72
- european: G06F7/72M
Application number: EP19930402865 19931126
Priority number(s): IL19920103921 19921130; IL19930104753 19930216; IL19930106923 19930906

Also published as:
US 5513133 (A1)
J P7253949 (A)
E P0601907 (A3)
E P0601907 (B1)

Cited documents:
US 5101431
E P0502782
E P0531158

Abstract of **EP0601907**

A compact synchronous microelectronic peripheral machine for standard microprocessors with means for proper clocking and control, has as-essential elements: three main subdivided, switched and clocked shift registers, B, S, and N; two only multiplexed serial/parallel multipliers; borrow detectors, ancillary subtractors and adders; delay registers and switching elements; all of which embody a totally integrated concurrent and synchronous process approach to modular multiplication, squaring, and exponentiation. A method for carrying out modular multiplication, wherein the multiplicand A , the multiplier B and the modul, N, comprise m characters of k bits each, the multiplier not being greater than the modulus, is also described, wherein the multiplicand can be much larger than the modulus. It is demonstrated how the device can be used as a large number processor in the normal field of numbers.

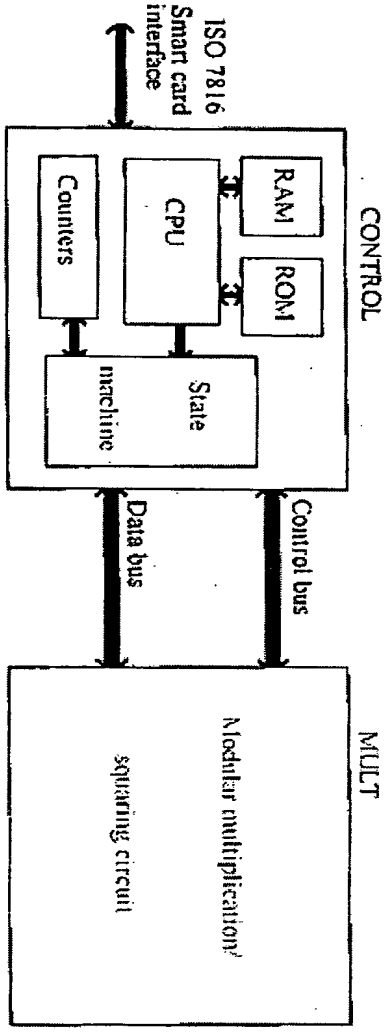

Fig. 1

(19) BUNDESREPUBLIK DEUTSCHLAND

DEUTSCHES PATENT- UND MARKENAMT

(12) Übersetzung der europäischen Patentschrift

(97) EP 0 601 907 B1

(10) DE 693 29 929 T 2

(51) Int. Cl.[7]: G 06 F 7/72

(21) Deutsches Aktenzeichen: 693 29 929.0
(95) Europäisches Aktenzeichen: 93 402 865.5
(96) Europäischer Anmeldetag: 26. 11. 1993
(97) Erstveröffentlichung durch das EPA: 15. 6. 1994
(97) Veröffentlichungstag der Patenterteilung beim EPA: 14. 2. 2001
(47) Veröffentlichungstag im Patentblatt: 27. 9. 2001

(30) Unionspriorität:

10392192	30. 11. 1992	IL
10475393	16. 02. 1993	IL
10692393	06. 09. 1993	IL

(73) Patentinhaber:
M-Systems Flash Pioneers Ltd., Kfar Saba, IL

(74) Vertreter:
Koepe, Fiesser & Partner Patentanwälte, 81245 München

(84) Benannte Vertragstaaten:
AT, CH, DE, DK, FR, GB, IT, LI, NL, SE

(72) Erfinder:
Gressel, Carmi David, Mobile Post Negev 85530, IL; Hendel, David, Raanana, IL; Dror, Itai, Beer-Sheva, IL; Hadad, Isaac, Beer-Sheva 84434, IL; Azari, Benjamin, Omer 84965, IL

(54) Mikroelektronische Kompaktanlage zum Ausführen modulärer Multiplizierung und Potenzierung mit grossen Operanden

DE 693 29 929 T 2

DE 693 29 929 T 2

14.09.01

0601907

Die vorliegende Erfindung bezieht sich auf die modulare Verarbeitung einer großen Zahl Primzahlen im Galois-Feld und auch von zusammengesetzten konstanten Primärkoeffizienten. Die Erfindung bezieht sich insbesondere auf eine Vorrichtung zur Implementierung modularer Multiplikationen / Potenzierungen von großen Zahlen, was zur Durchführung der Operationen geeignet ist, die für die Zugriffsberechtigungsprüfung von Geheimverschlüsselungen mit öffentlichem Schlüssel und Verschlüsselungsprotokollen wesentlich ist, die in zumutbarer Verarbeitungszeit nicht mit kleinen Mikroprozessoren durchgeführt werden können.

Die vorliegende Erfindung bezieht sich auf die Hardware-Implementierung eines Verfahrens, bekannt als „das überlappte Montgomery modulare Multipräzisions-Multiplikationsverfahren", das häufig in auf Verschlüsselungssoftware ausgerichteten Systemen eingesetzt wird. Es wird ein einzigartiges neuartiges Verfahren zur Beschleunigung der modularen Potenzierung dargeboten; und es werden vitale Nachweise zur Vereinfachung der Architektur und der Ausweitung des Einsatzes der Vorrichtung auf eine große Anzahl von Berechnungen im normalen Zahlenbereich eingesetzt.

Das grundlegende Verfahren ist eines der drei veröffentlichten, damit zusammenhängenden Verfahren zur Durchführung der modularen Multiplikation mit Montgomerys Methodologie. [P. L. Montgomery, „Modulare Multiplikation ohne Versuchsdivision", *Berechnungsmathematik*, Band 44, pp. 519-521, 1985], nachstehend als „Montgomery" bezeichnet [S. R. Dusse und B. S. Kaliski Jr., „Eine kryptographische Bücherei für Motorola DSP 56000" *Pro Eurocrypt* 90, Springer Verlag, Berlin, 1990], nachstehend als „Dusse" bezeichnet.

In dieser Hardware-Implementierung wurden Sicherheitsmechanismen und „fliegende" Additionen, Subtraktionen und Bewegungen hinzugefügt; Prozesse, deren Gesamtausgabe irrelevant sein könnte, wurden entfernt; ein relativ einfach zu implementierendes siliconartiges Design wurde erfunden und integriert, um dem internen Daten-/Adress-Bus als abhängiger Rechner an praktisch jede beliebige 8, 16 oder 32-bit Zentralprozessoreinheit (CPU) angehängt zu werden.

Aufgrund des einfach synchronisierten Umschaltdesigns kann die Multiplikations- / Quadrierungsmaschine bei Taktgeschwindigkeiten laufen, die um mehrere Male schneller sind als die mit einem CPU derzeitig erreichbaren Geschwindigkeiten, der On-Board nicht flüchtige Speichervorrichtungen unterstützt. Dieses Verfahren erfordert keine Veränderungen des Designs der Speicherarchitektur des CPU, wie dies bei Implementierungen mit parallelen Multiplizierwerken und Speichern mit Dualport zur schnellen modularen Multiplikation von großen Zahlen im Schaltkreis von Philips der Fall ist. [Philips Components, „83C852, gesicherter 8-bit Mikrokontroller für Anwendungen mit bedingtem Zugang", Eindhoven, August 1990], nachstehend als „Philips" bezeichnet.

Die wesentliche Architektur besteht aus einer Maschine, die in irgendein beliebiges Mikrocontroller-Design integriert und in den Speicher konvertiert werden kann; während parallel mit dem Kontroller gearbeitet wird, der ständig Befehle und Operanden laden und die endgültige Antwort dann entladen und übertragen muß.

Die einzigartige Lösung verwendet nur zwei serielle / parallele Multiplikationswerke und einen kompletten seriellen verknüpften Ansatz, der Siliconbereich spart. Durch den Einsatz aktueller bekannter Technologien ermöglicht sie die Integration der kompletten Lösung, unter Einschluß eines Mikrocontrollers mit Speichern bis zu einem 4 Mal 4 Mal 5 Mal 0,3 mm mikroelektronischem Schaltkreis, der den ISO-Normen genügen kann [International Organisation for Standardization, „Identifikationskarten - Karten mit integriertem Schaltkreis, ISO 7816:

Teil 1 - ISO 7816-1 „Physikalische Merkmale", 1987

Teil 2 - ISO 7816-2, „Abmessungen von Kontaktstellen", 1988

Teil 3 - ISO/IEC 7816-3 „Protokolle für elektronische Signale & Übertragungen", 1989]

nachstehend bezeichnet als „ISO 7816".

Die Erfindung ist auf die Architektur dieser Lösung ausgerichtet, basierend auf mathematischen, von Montgomery veröffentlichten Innovationen, mit mehreren Modifikationen und Verbesserungen, und es werden nicht naheliegende Verfahren zur Reduzierung der für die modulare Potenzierung notwendigen Zeit um etwas mehr als die Hälfte der erforderlichen Zeit bei Einsatz von bekannten Verarbeitungsverfahren und des Montgomery-Verfahrens dargelegt.

Definitionen, Allgemeine Prinzipien und Verfahren

Die Erfindung wird in der folgenden Beschreibung unter Einsatz der allgemeinen, nachstehend beschriebenen Prinzipien und Verfahren erläutert.

Zur modularen Multiplikation im Bereich der Primärzahlen und der zusammengesetzten Primärzahlen definieren wir A und B als Multiplikanden und den Multiplikator, und N als den konstanten Koeffizienten, der üblicherweise größer ist als A oder B. N kann in einigen Fällen kleiner sein als A. Wir definieren A, B und N als m . k = n Bit lange Operanden. Jede k Bit Gruppe wird Zeichen genannt. Dann sind A, B und N jeder m Zeichen lang. Zur Erleichterung der Verfolgung der ersten Implementierung und bei der schrittweisen Erläuterung des Verfahrens nehmen wir an, daß A, B und N 512 Bits lang sind (n = 512); nehmen wir an, daß k aufgrund der kosteneffektiven Länge der Multiplikatoren 32 Bits lang ist und m = 16 die Zahl der Zeichen in einem Operanden und ebenfalls die Zahl der Iterationen in einer Quadrierungs- oder Multiplikationsschleife mit einem 512 Bit Operanden ist. Es ist klar, daß alle Operanden ganze Zahlen sind.

Wir verwenden das Symbol = , um die Kongruenz von modularen Zahlen anzugeben, zum Beispiel 16 = 2 mod. 7, und wir sagen, 16 ist kongruent zu 2 Modulo 7, da 2 der Rest ist, wenn 16 durch 7 dividiert wird. Wenn wir schreiben Y mod., N = X; können sowohl Y als auch X größer sein als N; allerdings werden bei positivem X und Y die Reste identisch sein. Anzumerken ist ebenfalls, daß die Kongruenz einer negativen ganzen Zahl Y Y + uN ist, wobei N der Modulus ist, und falls die Kongruenz von Y geringer als N ist, wird u die kleinste ganze Zahl sein, was ein positives Ergebnis ergibt.

Wir verwenden das Symbol ¥ zur Bezeichnung der Kongruenz in einem engeren Sinne. Während der hierin beschriebenen

Prozesse ist ein Wert häufig entweder der gewünschte Wert oder gleich dem gewünschten Wert plus dem konstanten Koeffizienten. Zum Beispiel kann X ¥ 2 mod. 7 X gleich 2 oder 9 sein. Wir sagen, X hat eine begrenzte Kongruenz mit 2 mod. 7.

Wenn wir schreiben X = A mod. N, definieren wir X als den Rest von A, dividiert durch N, zum Beispiel 3 = 45 mod. 7.

In der Zahlentheorie ist das modulare multiplikative Gegenteil ein grundlegendes Konzept. Zum Beispiel wird das modulare multiplikative Gegenteil von X als X^{-1} geschrieben, das durch XX^{-1} mod. N = 1 definiert wird. Wenn X = 3 und N = 13, dann ist X^{-1} = 9, das heißt, der Rest von 3 . 9 dividiert durch 13 ist 1.

Die Akronyme MS und LS werden zur Bezeichnung der größten und der kleinsten Signifikanten bei der Referenzierung von Bits, Zeichen und fallen Operandenwerten eingesetzt.

In dieser gesamten Spezifikation bezeichnet N sowohl den Wert N als auch den Namen des Umschaltzahlwerkes, das N enthält. A und N sind während einer ganzen Potenzierung konstante Werte. A ist der Wert der Zahl, die zu potenzieren ist. Während der ersten Iteration einer Potenzierung ist B gleich A. B ist ebenfalls der Name des Zählwerks, in dem sich der kumulierte Wert, der letztendlich dem gewünschten Ergebnis der Potenzierung entspricht, befindet. S bezeichnet einen temporären Wert und ebenfalls das Zahlwerks, in dem ¥ von S gespeichert ist. S(i-1) bezeichnet den Wert von S zu Beginn der i. Iteration: S_0 bezeichnet das LS Zeichen eines S(i)ten Wertes.

Wir beziehen uns auf das (später definierte) Verfahren $\wp$(A . B)N als Multiplikation im $\wp$-Bereich oder manchmal einfach eines Multiplikationsvorgangs.

Andere Symbole sind diejenigen, die auch üblicherweise in der Arithmetik verwendet werden:

Modulare Multiplikation nach Montgomery

In einem klassischen Ansatz zur Berechnung einer modularen Multiplikation A . B mod. N wird der Rest des Produkts A . B durch einen Divisionsprozeß errechnet. Die Implementierung eines Divisionsprozesses ist schwieriger durchzuführen als ein Multiplikationsprozeß.

Durch Einsatz des modularen Reduktionsverfahrens nach Montgomery wird die Division im wesentlichen durch Multiplikationen ersetzt, die vorausberechnete Konstanten verwenden.

Die Funktion $\wp(A \cdot B)N$ nach Montgomery führt eine Multiplikations-Modulo N des Produkts A . B im $\wp$-Bereich durch. Die Wiedergewinnung aus dem $\wp$-Bereich zurück in den normalen modularen Bereich wird durch Darstellung von $\wp$ im Ergebnis von $\wp(A \cdot B)N$ und einer vorausberechneten Konstante H dargestellt. Falls nun $P = \wp(A \cdot B)N$ ist, dann ist $\wp(P \cdot H)N = A \cdot B \bmod N$; dadurch wird eine normale modulare Multiplikation in zwei $\wp$-Bereichs-Multiplikationen durchgeführt.

Das Ziel effizienter modularer Reduktionsverfahren ist die Verhinderung einer Serie von Multiplikations- und Divisionsvorgängen mit Operanden, die n und 2n Bit lang sind, durch die Durchführung einer Serie von Multiplikationen, Additionen und Subtraktionen mit Operanden, die n Bits lang sind und die ein Endergebnis erzielen, das ein Maximum an n Bit lang ist. Zur Darstellung der Lehre nach Montgomery beobachten wir, daß bei vorgegebenem A, B und ungeradem N (diese ungeraden konstanten Faktoren sind immer entweder einfach oder eine

Komponente großer Primzahlen) gibt es immer ein Q, so daß A . B + Q . N in einer Zahl resultiert, dessen n LS Bits Null sind oder

$$P \cdot 2^n = A \cdot B + Q \cdot N.$$

Dies bedeutet, daß wir einen 2n Bits langen Ausdruck haben, dessen N LS Bits Null sind.

Lassen wir nun $I \cdot 2^n = 1 \bmod N$ sein (I existiert für alle ungeraden Ns). Die Multiplikation beider Seiten der vorstehenden Gleichung durch I ergibt die folgende Kongruenz:

von der linken Seite der Gleichung:

$P \cdot I \cdot 2^n = N$ (Es sei daran erinnert, daß $I \cdot 2^n = 1 \bmod N$ ist)

und von der rechten Seite:

$A \cdot B \cdot I + Q \cdot N \cdot I = AB \cdot I \bmod N$ (Es sei daran erinnert, daß $Q \cdot N \cdot I = 0 \bmod N$ ist),

daher

$$P = A \cdot B \cdot I \bmod N$$

Leider bedeutet dies auch, daß ein parasitärer Faktor I jedes Mal dann eingeführt wird, wenn die Multiplikation im $\wp$-Bereich durchgeführt wird.

Wir definieren den $\wp$ Operator derart, daß:

$$P = A \cdot B \cdot I \bmod N = \wp(A \cdot B)N$$

und wir nennen dies „Multiplikation von A Mal B im $\wp$-Bereich".

Die Wiedergewinnung vom $\wp$-Bereich wird durch Anwendung von $\wp$ auf P . H berechnet, was ergibt:

$\wp(P \cdot H)N = A \cdot B \bmod N;$

Wir können den Wert von H durch Ersatz von P in der vorstenenden Kongruenz ableiten. Wir finden:

$\wp(P \cdot H)N = (A \cdot B)\ (H)\ (I) \bmod N;$

(siehe, daß A . B . I ← P; H ← H; [← und jeder Multiplikationsvorgang führt einen Parasiten I ein]

Falls H mit dem vielfachen Gegenteil von I^2 kongruent ist, dann ist die Kongruenz gültig, daher

$H = I^{-2} \bmod N = 2^{2n} \bmod N$

(H ist eine Funktion von N, und wir nennen es den Parameter H)

Zur Darstellung des $\wp$-Operators auf A . B, führen wir das folgende Verfahren unter Einsatz der vorausberechneten Konstante J fort:

1) X = A . B

2) Y = (X . J) mod. 2^n (es sind nur die n LS Bits notwendig)

3) Z = X + Y . N

4) S = Z / 2^n (Die Anforderung an J ist, daß es Z zwingt, durch 2^n teilbar zu sein)

5) P ¥ S mod. N (N ist von S zu subtrahieren, falls S ≥ N ist)

Schließlich, bei Schritt 5):

$P \neq \mu(A \cdot B)N$

[nach Subtraktion von N, falls nötig:

$P = \mu(A \cdot B)N]$

Unter Fortführung des obigen:

$Y = A \cdot B \cdot J \bmod. 2^n$ (unter Einsatz nur der n LS Bits)

und

$Z = A \cdot B + (A \cdot B \cdot J \bmod. 2^n) \cdot N$

Damit Z durch 2^n (die n LS Bits von Z müssen Null sein) teilbar ist, muß folgende Kongruenz existieren:

$[A \cdot B + (A \cdot B \cdot J \bmod. 2^n) \cdot N] \bmod. 2^n = 0$

Damit diese Kongruenz existiert, muß $N \cdot J \bmod. 2^n$ kongruent zu -1 sein oder:

$J = -N^{-1} \bmod. 2^n$

und wir haben die Konstante J gefunden.

Daher ist J eine vorausberechnete Konstante, die nur eine Funktion von N ist und naheliegenderweise müssen wir immer das positive J auswählen, das kleiner als N ist.

Daher verwendet das aufgezeigte Verfahren, wie dem Fachmann einleuchten wird, drei Multiplikationen, eine Summenbildung und ein Maximum an Subtraktionen für das vorgegebene A, B, N und eine vorausberechnete Konstante: wir erhalten $\mu(A \cdot B)N$. Unter Verwendung dieses Ergebnisses, desselben Verfahrens und einer vorausberechneten Konstante H (einer Funktion des Moduls N), sind wir in der Lage, $A \cdot B \bmod. N$ zu finden. Da A gleich B sein kann, kann dieser Operator als

eine Einheit zur Quadrierung oder Multiplikation in der modularen Arithmetik verwendet werden.

Überlappte Modulare Multiplikation nach Montgomery

Im vorstehenden Abschnitt wurde ein Verfahren zur modularen Multiplikation aufgezeigt, das Multiplikationen von Operanden verlangt, die alle n Bits lang wären, und Ergebnisse, die 2n + 1 Bits Speicherplatz erforderten.

Durch den Einsatz von Montgomerys überlappter Reduktion (gemäß des vorgenannten Dokuments von Dussel) ist es möglich, die Multiplikationsvorgänge mit kürzeren Operanden, Zählwerken und Hardware-Multiplikatoren durchzuführen, was die Implementierung einer elektronischen Vorrichtung mit relativ wenigen logischen Gattern ermöglicht.

Bei Einsatz eines k Bit Multiplikators ist es angebracht, Zeichen von K Bit Länge zu definieren; es gibt m Zeichen in n; das heißt m . k = n.

J_0 wird das LS Zeichen von J sein.

Daher

$$J_0 = -N_0^{-1} \text{ mod. } 2^k \quad (J_0 \text{ existiert als N ist ungerade})$$

Dann wird unter Einsatz von Montgomerys überlappter Reduktion $\wp(A \cdot b)N$ in m Iterationen unter folgender Anfangsbedingung und unter Fortführung der Schritte (1) bis (5) dargestellt. Der Verlauf der Erfindung folgt diesen Schritten auf zusammenfallende Weise.

Zu Beginn S(0) = 0 (der Y Wert von S zu Beginn der ersten Iteration).

Für i = 1, 2 ... m

(1)$X = S(i-1) + A_{i-1} \cdot B$ (A_{i-1} ist das i-1. Zeichen von A; S(i-1) ist der Wert von S zu Beginn der i. Iteration)

(2)$Y_0 = X_0 \cdot J_0 \bmod 2^k$ (Die LK k Bits des Produkts $X_0 \cdot J_0$)

(Das Verfahren verwendet und berechnet nur die k LS Bits, zum Beispiel den geringsten Signifikanten 32 Bits).

(3)$Z = X + Y_0 \cdot N$

(4)$S(i) = Z/2^k$ (Die k LS Bits von Z sind immer 0, daher ist Z immer durch 2^k teilbar. Diese Division ist gleichbedeutend mit einer k Bit rechten Versetzung, da die LK k Bits von Z alle Null sind; oder, wie im Verlauf gesehen werden wird, die LK k Bits von Z werden einfach unbeachtet gelassen.

(5)$S(i) = S(i) \bmod N$ (N ist von solchen S(i)s zu subtrahieren, die größer sind als N).

Schließlich, bei der letzten Iteration (nach der Subtraktion von N, falls notwendig), $C = S(m) = \wp(A \cdot B)N$. Um $F = A \cdot B \bmod N$ abzuleiten, müssen wir die Berechnung $\wp(C \cdot H)N$ im $\wp$-Bereich durchführen.

Nun beweisen wir, daß S(i) für alle S(i)s kleiner ist als 2N (nicht der in Montgomery Beweisführung inbegriffen).

Wir beobachten, daß bei Operanden, die im Verfahren verwendet werden:

$S(i-1) < N; \quad B < N \quad$ und $A_{i-1} < 2^k$

(Die ersten beiden Ungleichheiten bleiben, da zu Beginn einer Iteration N von S(i-1) und B subtrahiert wird, wenn sie entweder größer als oder gleich N waren. Die dritte

Ungleichheit bleibt, da 2k eine k + 1 Bit lange Zahl ist, deren MS Bit „1" ist, während A_{i-1} ein k Bit langer Operand ist).

Per Definition:

$S(i-1) = Z/2^k$ (Der Wert von S am Ende des Verfahrens vor einer möglichen Subtraktion)

Ersatz im obigen Gleichungssatz:

$$Z = S(i-1) \times A_{i-1} \cdot B + (X_0 \cdot J_0 \bmod 2^k) N$$

Anzumerken ist, daß bei Berücksichtigung des maximale Wertes jedes Elements in der vorstehenden Gleichung wir die Ungleichheit von Z erhalten:

$$Z < (N - 1) + (2^k - 1)(N - 1) + (2^k - 1) \cdot N = 2^k N + 2^k N - N - 2^k$$

und sicherlich

$$Z < 2^k N + 2^k N$$

Nun werden beide Seiten der Ungleichheit durch 2^k dividiert:

$$Z/2^k < N + N$$

und wir haben bewiesen, daß eine Subtraktion von N alles ist, was jemals zur Korrektur eines S(i) oder eines B notwendig ist.

Beispiel I

eine überlappte modulare Multiplikation:

Die folgenden Berechnungen können leicht mit einem Handrechner überprüft werden, der über einen hexadezimalen Modus verfügt. Unter Verwendung des hexadezimalen Format nehmen wir an:

N = a59 (der konstante Koeffizient), A = 99b (der Multiplikator), B = 5c3 (der Multiplikand), n = 12, (die Bit Länge von N), k = 4, (die Größe in Bits des Multiplikators and ebenfalls die Größe eines Zeichens) und m = 3, da n = k . m.

J_0 = 7 da 7 . 9 = -1 mod. 16 und H = $2^{2\cdot12}$ mod. a59 = 44b.

Das erwartete Ergebnis ist F = A . B mod. N = 99b . 5c3 mod. a59 = 375811 mod. a59 = 220_{16}

Zu Beginn: S(0) = 0

Stufe 1 $X = S(0) + A_0 . B = 0 + b . 5c3 = 3f61$
$Y_0 = X_0 . J_0$ mod. $2^k = 7$
$Z = X + Y_0 . N = 3f61 + 7 . a59 = 87d0$
$S(1) = Z / 2^k = 87d$ (welches kleiner als N ist)

Stufe 2 $X = S(1) + A_1 . B = 87d + 9 . 5c3 = 3c58$
$Y_0 = X_0 . J_0$ mod. $2^k = 8 . 7$ mod. $2^4 = 8$
$Z = X + Y_0 . N = 3c58 + 52c8 = 8f20$
$S(2) = Z / 2^k = 8f2$ (welches kleiner als N ist)

Stufe 3 $X = S(2) + A_2 . B = 8f2 + 9 . 5c3 = 3ccd$
$Y_0 = d . 7$ mod. $2^4 = b$
$Z = X + Y_0 . N = 3ccd + b . a59 = aea0$
$S(3) = Z / 2^k = aea$. da S(3) > N.

S(3) = aea - a59 = 91

Daher C = $\wp$(A . B)N = 91_{16}

Die Wiedergewinnung aus dem $\wp$-Bereich wird durch Berechnung von $\wp$(C . H)N erreicht:

Auch hier, zu Beginn: S(0) = 0

Stufe 1 $X = S(0) + C_0 . H = 0 + 1 . 44b = 44b$

$Y_0 = d$

$Z = X + Y_0 \cdot N = 44b + 8685 = 8ad0$

$S(1) = Z / 2^k = 8ad$

Stufe 2 $X = S(1) + C_1 \cdot H = 8ad + 9 \cdot 44b = 2f50$

$Y_0 = 0$

$Z = X + Y_0 \cdot N = 2f50 + 0 = 2f50$

$S(2) = Z / 2^k = 2f5$

Stufe 3 $X = S(2) + C_2 \cdot H = 2f5 + 0 \cdot 44b = 2f5$

$Y_0 = 3$

$Z = X + Y_0 \cdot N = 2f5 + 3 \cdot a59 = 2200$

$S(3) = Z / 2^k = 220_{16}$,

was der erwartete Wert von 99b 5c3 mod. a59 ist.

Die Gültigkeit der Operation kann intuitiv verstanden werden, wenn wir realisieren, daß, wenn wir in jeder Stufe k LS Nullen außer Betracht lassen, wir im wesentlichen die n MS Bits mit 2^k multiplizieren. Dementsprechend ist in jeder Stufe das i. Segment des Multiplikators ebenfalls eine mit 2^{ik} multiplizierte Zahl, was ihr die gleiche Wertung wie S(i) verleiht.

Modulare Reduktion auf einer Montgomery-Maschine in einem Multiplikationsverfahren

Viele Verschlüsselungsverfahren, wie zum Beispiel der NIST Digital Signatures Standard oder die das Chinese Remainder Theorem (der chinesische Lehrsatz zur Restlehre) verwendende modulare Potenzierung erfordern die Reduzierung einer Zahl, die größer ist (häufig über zweimal so groß) als ein zweiter konstanter Koeffizient. Diese modularen Reaktionen konnen effizient in einer überlappten Multiplikation nach Montgomery unter Einsatz einer erfindungsgemäßen Maschine und einer nicht naheliegenden Erweiterung des Algorithmus nach Montgomery ausgeführt werden.

Anzumerken ist, daß in den voranstehenden Beispielen impliziert wurde, daß n, die Länge des konstanten Koeffizienten des Operanden, ebenfalls die genaue Länge von N war. Für gewöhnliche Potenzierungen und Multiplikationen wäre dies am effizientesten. In den Fällen jedoch, in denen eine Reduzierung der Größe notwendig ist, kann eine zweite Konstante, $I^{-1} = 2^n \bmod. N$ eingesetzt werden, die, wenn Montgomery mit der derart reduzierten Zahl multipliziert wird, eine Operation mit einer minimalen Reduktion ergibt. Diese Konstante I^{-1} kann mit demselben Mechanismus berechnet werden, der die Konstante H berechnet (siehe Abschnirt über die Berechnung des Parameters H), namlich durch Plazierung des Moduls N in den wichtigsten Teil des Divisoroperanden, so daß sein bedeutendstes „I" im bedeutendsten Bit des Divisorregisters verbleibt. Die Zahl der Verschiebungs- / Versuchs-Subtraktionen, muß naheliegenderweise nun n + I - L sein, wobei L die Zahl der relevanten Bits von N ist. Anzumerken ist, daß dieses I^{-1} ein L Bits langer Operand sein wird.

Zum Beweis dieser Prämisse wiederholen wir zunächst noch einmal, daß die Multiplikation nach Montgomery von A B mod. N, (ρ(YA . B)N) die Kongruenz A . B . I mod. N erzielt. Wenn wir B = I^{-1} zuordnen, dann

$$\rho(A \cdot I^{-1})N = A \cdot I^{-1} \cdot I \bmod. N = A \bmod. N.$$

Beispiel 2

eine überlappte Reduktion nach Montgomery:

Um eine Reduktion von t auf mod. q (t mod. q) zu demonstrieren, in der die Länge des Multiplikationsregisters, in dem t ursprünglich gespeichert ist, 24 Bits lang ist, größer ist als die Länge von q.

Nehmen wir eine Wortlänge (Große des Maschinenmultiplikators) von 8 Bit und die folgenden Testvariablen an:

n = 24: k = 8; t = 0a f5 9b; q = 2b 13; und

R = I^{-1} = 2^{24} mod. q = 141 d

Bei einer einfachen Divisionsberechnung wissen wir durch Vergleich, daß t mod. q = 5c8 ist.

Anzumerken ist, daß die Reduzierung und die Wiedergewinnung in einer Multiplikation nach Montgomery durchgeführt werden:

Zu Beginn: S(0) = 0, A = t = 0a f5 9b, B = R = 141d, N = q = 2b 13

Stufe 1 X = S(0) + A_0 . B = 0 + 9 - 141d = c2d 8f
Y_0 = X_0 . J_0 mod. 2^k = 8f . e5 mod. 2^8 = eb
Z = X + Y_0 . N = c2d 8f + eb . 2b 13 = 33 b8 00
S(1) ¥ Z / 2^k mod. N = 33 b8, welches großer ist als N
S(1) = 33 b8 - 2b 13 = 8 a5

Stufe 2 X = S(1) + A_1 . B = 8 a5 + f5 . 141d = 13 48 66
Y_0 = X_0 . J_0 mod. 2^k = 66 . E5 mod. 2^8 = 3e
Z = X + Y_0 . N = 13 48 66 + 3e . 2b 13 = 1d b7 00
S(2) = Z / 2^k mod. N = 1d b7

Stufe 3 X = S(2) + A_2 . B = 1d b7 + 0A . 14 1d = e6 d9
Y_0 = d9 . e5 mod. 2^8 = 1d
Z = X + Y_0 . N = e6 d9 + 1d . 2b 13 = 5 c8 00
S(3) = Z / 2^k mod. N = 5 c8,

Und t mod. q = 5c8, wie zuvor berechnet.

Potenzierung

Die folgende Ableitung einer Sequenz [D. Knuth, *Die Kunst des Programmierens von Computern*, Band 2: seminumerische Algorithmen, Addison-Wesley, Reading Maß., 1981], nachstehend als „Knuth" bezeichnet, erläutert eine Sequenz von Quadraturen und Multiplikationen, die eine modulare Potenzierung implizieren.

Unter der Annahme, daß wir die Konstanten im obigen Abschnitt vorausberechnet haben und daß unsere Vorrichtung im ℘-Bereich sowohl Quadrieren als auch Multiplizieren kann, möchten wir folgendes berechnen:

$$C = A^E \text{ mod. } N$$

Lassen wir E(j) den j. Bit in der binären Darstellung des Exponenten E angeben, beginnend mit dem MS Bit, dessen Index I ist und abschließend mit dem LS Bit, dessen Index q ist, konnen wir wie folgt potenzieren:

a) B = A

BEI j = 2 BIS q

a) B ¥ ℘(B . B)N

b) B ¥ ℘(B . H)N (die Schritte a und b entsprechen B ¥ B^2 mod. N)

WENN E(j) = 1 DANN

a) B ¥ ℘(B . A)N

b) B ¥ ℘(B . H)N (die Schritte a und b entsprechen den B ¥ B . A mod. N)

Im Übergang von jedem Schritt zum nächsten wird N von B subtrahiert wann immer B größer als oder gleich N ist.

Nach der letzten Iteration ist der Wert B ¥ bis A^E mod. N.

Es gibt effizientere systemgebundene Protokolle, die mit der beschriebenen Schaltung zur Durchführung der modularen Potenzierung verwendet werden können; wir nennen zwei Verschlüsselungsprotokolle, bei der das hierin beschriebene Verfahren die Geschwindigkeit der Potenzierung häufig verdoppeln wird. Im RSA Verfahren [R. L. Rivest und andere, „Ein Verfahren zum Erhalt digitaler Unterschriften und Geheimverschlüsselungen mit öffentlichem Schlüssel", Kommentar zum ACM, Band 21, 120 - 126, 1978], nachstehend bezeichnet als „RSA" und dem Diffle-Hellman Protokoll [W. Diffle und M. E. Hellman, „Neue Richtungen bei der Verschlüsselung", IEEE Trans. on Inform. Theorie, Band IT-22, 644 - 654, 1976], nachstehend als „Diffle-Hellman" bezeichnet, werden die meisten der schwierigen Potenzierungen durch Einsatz eines konstanten Exponenten durchgeführt. Das Verfahren des folgenden Abschnitts (ein effizientes Verfahren für eine Wiedergewinnung von einer Potenzierung eines ρ-Bereichs) verringert die Berechnungszeit für jene Berechnungen, bei denen ein konstanter Exponent verwendet wird. Wenn dieses Verfahren eingesetzt wird, entfallen die Schritte b) im beschriebenen Potenzierungsverfahren (alle ρ(B . H)N Multiplikationen), und der endgültige Wert von B wird nach der q. Iteration der Potenzierung im ρ-Bereich nach Montgomery mit einer vorausberechneten Konstanten T multipliziert.

Denjenigen, die in die Implementierung eingebunden sind, leuchtet es ein, daß es für vollständige RSA-Unterschriften mit dieser Schaltung unter Verwendung des Chinese Remainder Theorem [beschrieben im vorgenannten Artikel von Knuth] möglich ist, eine weitere Reduzierung um mehr als 70% der Berechnungszeit vorzunehmen.

Ein effizientes Verfahren für eine Wiedergewinnung von einer $\wp$-Bereichs-Potenzierung

Das Quadrierungs- und Multiplikationsprotokoll des vorstehenden Absatzes kann verbessert werden, und es ist möglich, die Zahl der $\wp$-Bereichs-Multiplikationen während der iterativen Sequenz durch Einführung einer neuen vorausberechneten Konstanten T zu reduzieren, die eine Funktion des konstanten Koeffizienten N und des Exponenten E ist.

$$T = (2^{\Sigma})^{\Sigma} \text{ mod. } N = (I^{-1})^{\Sigma} \text{ mod. } N$$

Wo $\Sigma = 2^{q-1} + E \text{ mod. } 2^{q-1}$

und

q ist die Zahl der relevanten Bits in E (unter Wegfall aller voranstehenden Nullen)

Die modulare Potenzierung kann nun berechnet werden mit der Sequenz:

Zu Beginn: B = A

BEI j = 2 BIS q

B ¥ $\wp$(B . B)N

WENN E(j) = 1 DANN

B ¥ $\wp$(B . A)N

ENDE FÜR

B ¥ $\wp$(B . T)N

Nehmen wir wieder an, daß für jeden Übergang von einem Schritt zum nächsten N von B subtrahiert wird, wann immer B größer als oder gleich N ist.

Anzumerken ist wiederum, daß jede Multiplikation im $\wp$-Bereich einer modularen Multiplikation desselben Faktors mit L entspricht, zum Beispiel $\wp(X \cdot Y) = X \cdot Y \cdot I \bmod N$.

Beispiel 3

Dieses Beispiel demonstriert den Einsatz von T bei der Berechnung von $A^E \bmod N$ und macht Ts Definition deutlich.

Nehmen wir an n = 4 und E = 5 = 0101 . q (nach Wegfall von Es voranstehender Null) ist 3, daher:

E(1) = I; E(2) = 0; und E(3) = I

und T ist vorausberechnet:

$$T = (2^q)^E \bmod N = (I^{-1})^E \bmod N$$

$$\Sigma = 2^{q-1} + E \bmod 2^{q-1} = 2^{3-1} - 5 \bmod 2^{3-1} = 4 + 1 = 5$$

und daher:

$$T = I^5 \bmod N.$$

wie zu sehen ist, wenn

zu Beginn:

$$B = A$$

j = 2, E(2) = 0

$$B = \wp(B \cdot B)N = A^2 . I \bmod N$$

j = 3, E(3) = 1

$$B = \wp(B \cdot B)N = B^2 = A^4 \cdot I^2 \cdot I \bmod N$$

$$B = \wp(B \cdot A)N = A^4 . I^3 \cdot A \cdot I \bmod N$$

und schließlich:

$$B \text{ ¥ } \wp(B \cdot T)N = A^5 \cdot I^9 \cdot I^5 \cdot I \bmod N = A^5 \bmod N$$

Die Einführung des Parameters T kann vermieden werden, wenn die nachstehenden Schritte zur Berechnung von A^E befolgt werden:

Annehmend, daß wir die Konstante H nach Montgomery vorausberechnet haben und daß unsere Vorrichtung im $\wp$-Bereich sowohl Quadraturen als auch Multiplikationen berechnen kann, möchten wir folgende Berechnung anstellen:

$$C = A^E \bmod N$$

Lassen wir E(j) den j. Bit in der binären Darstellung des Exponenten E angeben, beginnend mit dem MS Bit, dessen Index I ist und abschließend mit dem LS Bit, dessen Index q ist, können wir wie folgt für ungerade Exponenten potenzieren:

```
A*¥ ℘(A . H)N
B = A*
BEI j = 2 BIS q-1
B ¥ ℘(B . B)N
WENN E(j) = 1 DANN
B ¥ ℘(B . A*)N
ENDE FÜR
B ¥ ℘(B . A)N
C = B
```

Beim Übergang von jedem Schritt zum nächsten wird N von B subtrahiert, wann immer B größer als oder gleich N ist.

Nach der letzten Iteration ist der Wert B ¥ bis $A^E \bmod N$, und C ist der endgültige Wert.

Bei geraden Exponenten könnte der letzte Schritt sein:

B ¥ ℘(B . 1)N anstatt von B ¥ ℘(B . A)N

Zur Klarstellung werden wir folgendes Beispiel verwenden:

$E = 1011 \rightarrow E(1) = 1,\ E(2) = 0;\ E(3) = 1;\ E(4) = 1;$

Um auf A^{1011} mod. N; q = 4 zu kommen

$A^* = \wp\ (A\ .\ H)N = AI^{-2}\ I = AI^{-1}$ mod. N

$B = A^*$

bei j = 2 bis q

$B = \wp(B\ .B)N$, was $A^2\ (I^{-1})^2\ .\ I = A^2 - I^{-1}$ ergibt

$E(2) = 0;$ $B = A^2\ .\ I^{-1}$

$j = 3$ $B = \wp(B\ .\ B)N = A^2\ (I^{-1})^2\ .\ I = A^4 - I^{-1}$.

$E(3) = 1$ $B = \wp(B.\ A^*)N = (A^4\ .\ I^{-1})\ (AI^{-1})\ .\ I = A^5 - I^{-1}$

$j = 4$ $B = \wp(B\ .\ B)N = A^{10}\ .\ I^2 - I = A^{-0}\ .\ I^{-1}$

Da E(4) eine ungerade Zahl war, erfolgt die letzte Multiplikation mit A, um den Parasiten I^1 zu entfernen.

$B = \wp(B\ .\ A) = A^{10}\ .\ I^{-1}\ .\ A\ .\ I = A^{11}$

$C = B$

Berechnung des Parameters H

Der Parameter H ist eine Konstante, die für Berechnungen im Bereich Montgomery von grundlegender Bedeutung ist. Bei Verwendung bestimmter Protokolle wird H eine Konstante sein, die auf einem größeren Computer vorausberechnet werden könnte, oder in anderen Fallen eine sinnvolle Konstante sein könnte, die in einer ersten Stufe ein Parameter für Berechnungen einer nützlicheren Konstante sein wird. Siehe vorstehender Abschnitt.

Bei allgemein üblichen Kommunikationen kann angenommen werden, daß H vorausberechnet wird, allerdings könnte es bei mehreren Protokollen, zum Beispiel bei der

Zugriffsberechtigungsprüfung einer Unterschrift bei einer Direktkommunikation in RSA notwendig sein, H mit dieser Vorrichtung zu berechnen, zum Beispiel der SmartCard.

Der Parameter H wird definiert als:

$$H = 2^{2n} \bmod N$$

Das bedeutet, daß H der Rest eines normalen Divisionsvorgangs ist, in dem eine Zeichenfolge, bei der auf ein MS Bit 2n LS Nullen folgen (a $2n + 1$ Bits langer Operand), durch die modulare Basis N dividiert wird.

Die binäre Division durch einen Divisor N eines Dividenden, bestehend aus einer „1" und einer Zeichenfolge von Nullen, ist gleichbedeutend mit der sequentiellen Versuchssubtraktion N, das heißt der Subtraktion von N vom restlichen Versuchsdividenden, wenn die signifikantesten $n+1$ Bits größer sind als N (siehe Beispiel).

Obwohl der Dividend $2n+1$ Bits lang ist, ist es naheliegend, daß der restliche Versuchsdividend, der durch eine Subtraktion beeinflußt wird, niemals länger als $n + 1$ Bits lang ist und die LS Ziffern Nullen sind.

Zum Beispiel:

H berechnen, wenn = $11_{10} = 1011_{11}$ (daher ist die Bit-Länge von N 4, das heißt n = 4)

Division, wie wir es manuell bei langen Divisionen der Basis 2 tun würden:

```
1 0111
1011 11 0000 0000
 1011          Erfolgreiche Subtraktion
 0101 0        ⇐ Ergebnis der ersten Runde
  101 1        Keine Subtraktion
```

101 00 ⇐ Ergebnis der zweiten Runde
10 11 Erfolgreiche Subtraktion
10 010 ⇐ Ergebnis der dritten Runde
1 011 Erfolgreiche Subtraktion
0 1110 ⇐ Ergebnis der vierten Runde
1011 Erfolgreiche Subtraktion

Ergebnis der 5. Runde (n+1)⇒ 0011 = H 3 Basis 10 = der Rest)

wo wir überprüft haben, daß H = 3_{10}.

Es gibt n + 1 Versuchssubtraktionen in einem H Divisionsverfahren. Anzumerken ist ebenfalls, daß der Versuchsdividend ebenfalls n + 1 Bits lang ist. Diese Sequenz der Subtraktionen wird in der Hardware gemäß der nachstehenden Beschreibung verfolgt.

Diese Erfindung betrifft ein mikroelektronisches Gerät und ein Verfahren zu seinem Einsatz zur Durchführung der modularen Quadrierung und der modularen Multiplikation eines Multiplikators durch einen Multiplikanden, jeweils gemäß Anspruch 1 und 18.

Die Ansprüche 2 bis 17 betreffen weitere Ausbildungsarten des Geräts gemäß Anspruch 1.

Diese Erfindung bezieht sich auf eine kompakte synchrone mikroelektronische Peripheriemaschine für Standardmikroprozessoren mit Mitteln für eigene Taktung und Kontrolle, deren wesentliche Elemente sind: drei unterteilte, geschaltete und getaktete Umschaltzählwerke B, S und N; zwei einzige serielle / parallele Multiplex-Multiplikationswerke; Entnahme-Sensoren, Hilfssubtraktionswerke und Addierwerke; Verzögerungszählwerke und Schaltelemente; die alle einen vollständig integrierte zusammenfallenden und synchronen Verfahrensansatz zu einer

modularen Multiplikation, Quadrierung und Potenzierung darstellen. Eine weitere Ausbildungsart implementiert eine einzigartige, nicht direkt ableitbare synchronisierte Hardware-Ableitung des Montgomery-Verfahrens, die zur modularen Hardwaremultiplikation, -quadrierung und -potenzierung ausgelegt ist. Alternativ ist es ebenfalls möglich, eine Ableitung des Verfahrens nach Montgomery als eine Vielzahl von simultanen seriellen Prozessen durchzuführen, das heißt, Multiplikationen, Subtraktionen, Additionen, gespeicherte Verzögerungen und eine Division durch 2^k. Die Prozesse werden parallel in dem Maße durchgeführt, wie serielle Prozesse darin aufgehen.

In einer weiteren Ausbildungsart it es möglich, eine Ableitung des Verfahrens nach Montgomery als eine Vielzahl von seriellen Prozessen zur modularen Multiplikation, Quadrierung und Potenzierung durchzuführen, unter Ausschluß des Einsatzes von weiten internen Bussen. Es ist weiterhin möglich, diese Ableitung ausreichend kompakt zu machen, damit sie auf einem Mikrochip hergestellt wird, gemäß den Normen ISO 7816 für tragbare Smart Cards, die die bekannte 1 Mikron Technologie verwenden.

Weiterhin wird davon ausgegangen, daß es möglich ist, eine Ableitung des Verfahrens nach Montgomery gemäß obiger Beschreibung als eine Vielzahl serieller Prozesse zur modularen Multiplikation, Quadrierung und Potenzierung durchzuführen, die durch irgendeinen beliebigen Mikroprozessor mit einem internen Bus kontrolliert werden kann, ohne seine grundlegende Architektur zu ändern und insbesondere ohne eine Veränderung des Designs der Speicher für den dualen Portzugang und mit relativ geringen Anforderungen an die Firmware.

Eine derartige Maschine kann den Mikrokontroller ebenfalls zur Regulierung der Kaskade von ρ-Bereichs-Sequenzen von Quadrierungen und Multiplikationen einsetzen, in denen der Exponent E nicht im MULT Block gespeichert zu werden braucht, so daß ein n-Bit langes Umschaltzählwerk gespart wird; die MULT-Kontrolle wird vereinfacht, während ein geringer zusätzlicher Mikrokontroller ROM-Coder erforderlich ist.

Gemäß einer weiteren Ausbildungsart der Maschine als Ergebnis des Ladens des A_1 Zählwerks mit den „fliegenden" Quadrierungsmultiplikanden, während das Register B rotiert, wird das Abladen durch den Mikrokontroller der vorherigen endgültigen Werte von B und / oder B-N ausgeschlossen, um das Zahlwerk A_i mit Zeichen B_i wiederaufzuladen. Dies schont RAM des Mikrocontrollers und eliminiert wenigstens n effektive Taktgeberzyklen bei jeder Quadrierungsiteration.

In einer weiteren Variante entfallen zwei Speicherzählwerke und getrennte serielle Subtraktionsvorgänge von einer direkten Implementierung des Verfahrens nach Montgomery. Dies erfolg durch Darstellung einer einzigen seriellen Erkennung auf $Z/2^K$ minus N zur Bestimmung, ob $Z/2^K$ größer als oder gleich N ist und anschließend bei nur einer seriellen Subtraktion kleiner werden kann als N Operanden.

In einer weiteren Ausbildungsart ist der Schaltkreis derart quasi-parallel synchronisiert, daß nur zwei Multiplikatoren zur Durchführung von drei simultanen Multiplikationsvorgängen eingesetzt werden. In einer Silikonimplementierung können serielle / parallele Multiplikatoren 40% des Silikonbereichs einnehmen. Bei Einsatz von zwei anstatt drei seriellen / parallelen Multiplikatoren verbleibt ausreichend Silikon zur Verdoppelung der Zahl der Zellen in den verbleibenden zwei

Multiplikatoren. Siese Verdoppelung der Größe der Multiplikatoren reduziert die Dauer des Prozesses bei einer 512 Bit Multiplikation um über 45%.

Es leuchtet ebenfalls ein, daß eine Maschine gemäß obiger Beschreibung mit einem digitalen Verzögerungsschaltkreis Delay3 (einem k Bit Umschaltzählwerk) zur Synchronisierung des seriellen Addierwerks von X mit dem seriellen Ergebnis des Multiplikators ML1, Y_0 . N eingesetzt werden kann, unter Ausschluß der doppelten Speicherung von Produkten oder einer Wiederholung einer seriellen / parallelen Multiplikation.

In einer weiteren Ausbildungsart gibt es zwei digitale Verzogerungsschaltkreise, Delay1 und Delay2 (zwei k Bit Umschaltzahlwerke), die zur Synchronisierung von drei seriellen Multiplikationen verwendet werden, in denen N ein Faktor ist, daß heißt B . A_1, X . J_0 und Y_0 . N.

Alternativ könnte eine Maschine gemäß obiger Beschreibung konstruiert werden, in der ein digitaler Verzögerungsschaltkreis Delay3 den Vorgang des seriellen / parallelen Multiplikators ML2 synchronisiert, so daß er zwei getrennte Multiplikationsvorgänge im Prozeßstrom durchführen kann, das heißt X . J_0 und Y_0 . N.

Es ist ebenfalls möglich, daß in einer solchen Maschine die Zählwerke S, B und N konfiguriert sind, um entweder n Bits oder n/2 Bits lang zu sein; wobei die Potenzierung über n/2 Längenmodule in etwas mehr als einem Achtel des effektiven Taktgerberzyklus abgeschlossen werden kann, der für n Bit lange Potenzierungen notwendig wäre.

In einer weiteren Ausbildungsart wird eine Maschine gemäß obiger Beschreibung dargeboten, die, wenn sie mit einem Original-Wiedergewinnungsfaktor T betrieben wird, die Zahlen der Multiplikationsvorgängen im P-Bereich bei einer

vollständigen Potenzierung der RSA-Unterschrift auf beinahe die Hälfte reduzieren.

Unter der Vorwegnahme notwendiger Vorausberechnungen konnte diese Maschine aufgrund des „fliegenden" Ladens des A Zählwerks, der „fliegenden Vorhersage" der Große der Inhalte des Zählwerks S und der „fliegenden" Synchronisierung des partiellen Operanden ebenfalls den vollständigen Multiplikationsvorgang $\wp(A \cdot B)N$ einer n Bit Zahl in nur $m(n+2k)$ effektiven Taktgeberzyklen ausführen.

Diese Erfindung umfaßt weiterhin eine Maschine gemäß obiger Beschreibung, die dieselben Zählwerke in derselben Maschine wie bei den Multiplikationen nach Montgomery verwendet, zu der ein kleiner Entnahme-Sensor-Schaltkreis und dem Kontrollmechanismus ein einfaches Addierwerk hinzugefügt wurden, das in einem zweiten Modus arbeitet und den Parameter H berechnet.

Es wird weiterhin vorweggenommen, daß jedes untergeordnete Verfahren und jedes Verfahren mit vorbestimmten Zahlen von Taktgeberzyklen ausgeführt werden, so daß eine Multiplikation im $\wp$-Bereich und / oder eine Quadrierung in bekannten Sequenzen von Taktgeberzyklen durchgeführt werden, was die Ausbildung einer vereinfachten Kontrolle ermöglicht, die in einer Kaskade von selbsterregenden Zählmechanismen ohne interne Bedingungszweige besteht.

Für jede der beschriebenen Maschinen (infra oder supra) könnte ein sogar noch weiter verbessertes Verfahren zur Durchfuhrung modularer Potenzierungen von $D = A^E \bmod N$ geliefert werden, bestehend aus den Schritten:

1. Speicherung des Exponenten E in einem Zählwerk des Computers

2. Laden des konstanten Koeffizienten in vorgenanntes Zählwerk N;

3. Setzen des vorgenannten Zählwerks S auf Null;

4. Durchführung eines Multiplikationsvorgangs

von $A^* = \wp(A \cdot H)_N$, während A der zu potenzierenden Operand ist und H ein vorausberechneter Parameter wie zuvor definiert.

5. Laden von A^* in das Basiszählwerk B.

6. Durchführen eine Quadrierungsvorgangs der Inhaltes des Zählwerks B.

7. Umschalten besagten übrigen Exponenten E;

8. Ignorieren aller seiner Null Bits, die dem ersten 1 Bit vorausgehen und Ignorieren des ersten 1 Bits besagten Exponenten E und für alle folgenden Bits, die die Vorgänge 9 bis 10 durchführen;

9. Für jedes einzelne der besagten E Bits, unabhängig davon, ob sie 0 sind oder 1, Ausführen der Vorgänge 4 und 5 der zuvor dargestellten Quadrierungsmethoden, bei denen sowohl der Multiplikand als auch der Multiplikator aus dem Zählwerk B stammen und in dem die aufeinander folgenden Zeichen des Multiplikators nach Montgomery vom Zählwerk B ins Zählwerk A_1 geladen werden.

10. Falls das aktuelle Bit des Exponenten E 1 ist, und nur dann, nach Durchführung des Vorgangs 9, Durchführung der Vorgänge 4 und 5 des zuvor hierin beschriebenen Multiplikationsvorgangs, in dem der Multiplikand der Inhalt des Zählwerks B und der Multiplikator die Basis A^* ist; und

11. nach Durchführung der Schritte 8 - 10 für alle Bits von E, Durchführung einer zusätzlichen Multiplikation des Zählwerks B durch die ursprüngliche Basis A und dann Speichern des Ergebnisses des letzten Vorgangs als $D \equiv A^E$ mod. N im Zählwerk B.

Ein weiterer Gegenstand der Erfindung (wiederum unter Einsatz jeglicher hierin (infra oder supra) beschriebenen Maschinen oder Verfahren) umfaßt ein Verfahren zur Durchführung herkömmlicher Multiplikationen von zwei Zahlen, deren durchschnittliche signifikante Länge n/2 Bits ist, umfassend die Durchführung der modularen Multiplikation besagter Zahlen durch das Multiplikationsverfahren gemäß Beschreibung wenigstens eines der hierin beschriebenen Verfahren (infra oder supra), in dem der konstante Koeffizient N eine n Bit Zahl ist, bestehend aus allen „1en" (fffffff ... fff), unter Gleichsetzung von J_0 mit 1 und Laden des Multiplikanden in B und Bearbeitung von A wie in besagtem Multiplikationsverfahren von Anspruch 1; N kann alle Einsen sein, entweder durch ein Vorlade-Zählwerk N mit allen Einsen oder durch Einstellen des Multiplexers, der N ausgibt, um eine Serie von „harten" Einsen auszugeben.

Es werden ein zusammenfallendes Verfahren und eine einzigartige Hardwarearchitektur dargeboten, um eine modulare Potenzierung ohne Division mit derselben Zahl an Vorgangen durchzuführen, die mit einer herkömmlichen Multiplikations- / Divisionsvorrichtung durchgeführt worden wäre, in dem eine herkömmliche Vorrichtung bei jedem Vorgang sowohl eine Multiplikation als auch eine Division durchführen würde. Eine Division ist üblicherweise ein nicht deterministisches Verfahren und gilt als schwieriger und zeitraubender als eine Multiplikation.

Die in dieser Erfindung realisierten Vorteile resultieren aus einer synchronisierten Sequenz serieller Prozesse, die zusammengeführt werden, um simultan (parallel) drei Multiplikationsvorgänge auf n Bit Operanden zu erreichen, unter Einsatz von zwei einfachen k Bit seriellen / parallelen Multiplikatoren in (n + 2k) effektiven Taktgeberzyklen.

Durch geeignete Synchronisierung und „fliegende" Erfassung und vorladende Operanden arbeitet die Maschine in einer deterministischen Weise, bei der alle Multiplikationen und Potenzierungen in einer vorbestimmten Zahl von Taktgeberzyklen ausgeführt werden. Bedingte Zweige werden durch lokale Erfassung und Ausgleichsvorrichtungen ersetzt, was eine Grundlage für den Kontrollmechanismus der einfachen Art bietet, die, wenn sie verfeinert wird, aus einer Serie selbsterregender Zähler in Kaskadenform bestehen kann.

Die Maschine stellt besonders geringe Anforderungen an den flüchtigen Speicher, da die Operanden für die gesamte Dauer des Vorgangs in die Maschine geladen und dort gespeichert werden, allerdings nutzt die Maschine die CPU, an die sie angeschlossen ist, zur Ausführung einfacher Ladungen und Abladungen und Befehlssequenzen der Maschine, während die Maschine ihre große Zahl an Berechnungen durchführt. Die Verarbeitungszeit der Potanzierung ist tatsächlich unabhängig von der CPU, die sie kontrolliert. In der Praxis sind keine Veränderungen der Architektur notwendig, wenn die Maschine an irgendeine beliebige CPU angeschlossen wird. Die Hardware-Vorrichtung ist selbstbeinhaltend und kann an jede CPU Bus angeschlossen werden.

Bei Einsatz dieser und zuvor patentierter sowie Verfahrenskontrollprotokolle des Standes der Technik wird das Mittel zur Beschleunigung des modularen Multiplikations-

und Potenzierungsverfahrens geliefert, zusammen mit Mitteln zur Vorausberechnung der notwendigen Konstanten.

Das Design der bevorzugten Ausbildungsarten der hierin beschriebenen Erfindung wurde kompaktiert und für den besonderen Zweck ausgelegt, einen modularen mathematischen Operator für Anwendungen mit Geheimverschlüsselungen mit öffentlichem Schlüssel auf portablen SmartCards darzubieten (die in Form und Größe mit den beliebten Magnetstreifenkredit- und -bankkarten identisch sind). Diese Karten werden in einer neuen Generation von Vorrichtungen mit Geheimverschlüsselungen mit öffentlichem Schlüssel zur Zugangskontrolle von Computern, Datenbanken und kritischen Installationen verwendet werden: zur Regulierung und Sicherung von Datenflüssen bei kaufmännischen, militarischen und häuslichen Transaktionen; zur Entschlüsselung codierter privater Fernsehsender, usw.

Es dürfte geschätzt werden, daß die Vorrichtung ebenfalls in Computer- und Faxterminals, Türschlösser, Verkaufsautomaten, usw. eingebaut werden kann.

Die beschriebene Hardware führt eine modulare Multiplikation und Potenzierung durch Anwendung des $\wp$-Operators in einem neuen einzigartigen Verfahren durch. Weiterhin kann die Quadrierung auf dieselbe Weise durch Anwendung auf einen Multiplikanden and einen Multiplikator durchgeführt werden, die gleich sind. Modulares Potenzieren impliziert eine Folge modularer Multiplikationen und Quadrierungen, und daher wird es durch ein Verfahren durchgeführt, das die wiederholte, geeignet kombinierte und ausgerichtete Anwendung der vorgenannten Multiplikations-, Quadrierungs- und Potenzierungsverfahren umfaßt. Jedoch wird hierin eine neue und verbesserte Weise der Durchführung der modularen Potenzierung weiter spezifiziert.

Das Verfahren zur Durchführung modularer Multiplikationen, bei denen der Multiplikand A, der Multiplikator B und der konstante Faktor jeder N m Zeichen von K Bits umfassen, wobei der Multiplikand und der Multiplikator nicht großer sind als der konstante Faktor, umfaßt die Schritte:

1 - Vorausberechnen eines Parameters H und wenigstens des am wenigsten signifikanten Zeichens J_0 eines anderen Parameters J gemäß nachstehender Definition und Laden von J_0 in ein k Bit Zählwerk.

2 - Laden des Multiplikanden B und des konstanten Faktors n in jeweilige Zählwerke von n Bit Länge, in denen n = m . k;

3 - Setzen eines n Bit langen Zählwerks S auf Null und

4 - Durchführung einer i Iteration m Male, in der i von Null bis m-1 ist, wobei jede i. Iteration die folgenden Vorgänge umfaßt:

a) Übertragung des i. Zeichens A_{i-1} auf den Multiplikanden A vom Zählwerkmittel A_i in das Speichermittel, das aus Zählwerkmitteln und Schaltermitteln ausgewählt ist;

b) Erzeugen des Wertes $X = S(i-1) + A_{i-1} \cdot B$, in dem S(i-1) der „aktualisierte" Wert von S gemäß hierin gegebener Definition ist, durch:

I. Rechtes Zyklusumschalten des Zählwerks B in die Multiplikationsmittel.

II. Serielles Multiplizieren von B mit A_{i-1}

II. Rechtes Zyklusumschalten des Koeffizienten-Zählwerks N

IV. Bestimmung des „aktualisierten" Wertes von S(i-1) als den im Zählwerk S gespeicherten Wert

nach der (i-1). Iteration, falls derselbe nicht größer ist als N, oder, falls er großer ist als N, durch

sein serielles Subtrahieren von N und Vorwegnahme der Ergebnisse als den „aktuellen" Wert

von S(i-1); und

v. Rechtes Zyklusumschalten des Zählwerks S und serielles Addieren des Wertes der Multipli-

kation A_{i-1} . B Bit Mal Bit zum „aktualisierten" Wert von S.

c) Multiplizieren des LS Zeichens von X, X_0 mit J_0 und Eingabe des Wertes X_0 . J_0 mod. 2^k in das Zählwerkmittel als Y_0 bei Verzögerung von N und X durch k Taktgeberzyklen:

d) Berechnung des Wertes $Z = X + Y_0$. N durch:

I Multiplizieren von Y_0 mit N durch eine verzögerte rechte Umschaltung des Zählwerks N

zusammenfallend mit der vorgenannten rechten Zyklusumschaltung davon, und

II Addieren von X zum Wert von Y_0 . N;

e) Ignorieren des geringsten signifikanten Zeichens von Z und Eingabe der restlichen Zeichen in das Zählwerk S unter Eingabe von $Z/2^k$ mit Ausnahme der letzten Iteration;

f) Vergleich von $Z/2^k$ mit N Bit per Bit zum Zwecke der Bestimmung des aktualisierten Wertes von S, S(i) in der hierin zuvor definierten Weise;

g) in der das i. Zeichen des Multiplikanden A_i zu jeder beliebigen Zeit während des vorgenanten Vorgangs in das Zählwerkmittel A geladen wird;

5) bei der letzten Iteration (m th) Ignorieren des geringsten signifikanten Zeichens von $Z/2^K$ und Eingabe der verbleibenden Zeichen in das Zählwerk B als den Wert von C ¥ $\wp$(A . B)N;

6) Wiederholen der Schritte 3) bis 4), in denen C oder C-N, falls C größer ist als N, durch B ersetzt wird und H durch A ersetzt wird, um P = $\wp$(C . H) mod. N zu berechnen; und

7) Vorwegnahme des Wertes $\wp$, der aus der letzten Iteration als das Ergebnis der Vorgänge A . B mod. N erhalten wird.

Es wird ebenfalls ein Verfahren zur Durchführung der modularen Potenzierung von $D = A^E$ mod. N beschrieben, das die folgenden Schritte umfaßt:

1) Laden der Zahl des Koeffizienten in das vorgenannte Zählwerk N;

2) Setzen des vorgenannten Zählwerks S auf Null;

3) Laden der zu potenzierenden Basis A in vorgenanntes Zahlwerk B;

4) Speichern des Exponenten B in ein Computerzählwerk;

5) Umschalten besagten übrigen Exponenten E;

6) Ignorieren aller seiner Null Bits, die dem ersten 1 Bit vorstehen und Ignorieren des ersten 1 Bits besagten Exponenten E und Durchführung der Vorgänge 7 bis 9 für alle folgenden Bits;

7) Für jedes einzelne besagte Bit, unabhängig davon, ob es 0 ist oder 1, Quadrieren des Inhalts des Zahlwerks B mit dem hierin dargelegten Multiplikationsverfahren, in dem die aufeinander folgenden Zeichen der Basis aus dem Zahlwerk B in das Zählwerk A_1 geladen werden.

8) Wenn das aktuelle Bit des Exponenten E 1 ist, und nur dann, Multiplizieren des Inhalts des Zählwerks B mit der Basis A nach Durchführung des Vorgangs 7), und

9) Nach jedem Quadrierungsvorgang nach Montgomery oder jedem Multiplikationsvorgang nach Montgomery Durchführung einer C . H Multiplikation $(\rho(C \cdot H))N$ nach Montgomery und

10) Nach Durchführung der Schritte 6 - 9 für alle Bits von E, Speichern des Ergebnisses des letzten Vorgangs als $D \leftarrow A^E$ mod. N im Zählwerk B.

Weiterhin wird ein Verfahren zur Durchführung einer modularen Potenzierung von $D = A^E$ mod. N beschrieben, die die Schritte umfaßt:

1) Laden der Zahl des Koeffizienten in das vorgenannte Zählwerk N;

2) Setzen des vorgenannten Zählwerks S auf Null;

3) Laden der zu potenzierenden Basis A in vorgenanntes Zählwerk B;

4) Speichern des Exponenten E in ein Computerzahlwerk und eines vorausberechneten Parameters T in einen CPU-Speicher;

5) Umschalten besagten übrigen Exponenten E

6) Ignorieren aller Null Bits davon, die dem ersten 1 Bit vorstehen und Ignorieren des 1 Bit besagten Exponenten E und Durchführung der Vorgänge 7 bis 8 für alle folgenden Bits:

7) für jedes einzelne der besagten Bits, unabhängig davon, ob es 0 ist oder 1, Durchführung der Vorgänge 4 und 5 des hierin dargelegten Multiplikationsverfahrens, in dem sowohl der Multiplikand als auch der Multiplikator die Basis A sind

und in dem die aufeinander folgenden Zeichen der Basis aus dem Zählwerk B in das Zählwerk A_i geladen werden;

8) Wenn das aktuelle Bit des Exponenten E 1 ist, und nur dann, nach Durchführung des Verfahrens 7, Durchführung der Vorgänge 4 und 5 des hierin dargelegten Multiplikationsverfahrens, in denen der Multiplikand der Inhalt des Zählwerks B ist und der Multiplikator die Basis A ist, und

9) Nach Durchführen der Schritte 7 und 8 für alle Bits von E, Durchführung einer zusätzlichen Multiplikation nach Montgomery des Zählwerks B mit dem Parameter T (ρ(B. T)N und dann Speichern des Ergebnisses des letzten Vorgangs als D ¥ A^E mod. N im Zählwerk B.

Parameter T wird definiert als $T = (2^n)^S$ mod. N, in dem

$S = 2^{q-1} + E$ mod. 2^{q-1} wie in der übergeordneten Anwendung im einzelnen erläutert.

Ein sogar noch weiter verbessertes Verfahren zur Durchführung der modularen Potenzierung von $D = A^E$ mod. N, das die Schritte umfaßt:

1) Speichern des Exponenten E in einem Computerzählwerk.

2) Laden der Zahl des Koeffizienten in vorgenanntes Zählwerk N;

3) Setzen des vorgenannten Zählwerks S auf Null;

4) Durchführen eines Multiplikationsvorgangs von $A^* = \rho$(A . R)N, während A der zu potenzierende Operand und R ein gemäß vorstehender Definition vorausberechneter Parameter ist.

5) Laden von A^* in das Basiszählwerk B.

6) Durchführung eines Quadrierungsvorgangs der Inhalte des Zählwerks B.

7) Umschalten des besagten übrigen Exponenten E.

8) Ignorieren aller Null Bits davon, die dem ersten 1 Bit vorstehen und Ignorieren des ersten 1 Bits des besagten Exponenten E und Durchführen des Vorgangs 9 bis 10 für alle folgenden Bits.

9) Für jedes einzelne der besagten E Bits, unabhängig davon, ob sie 0 sind oder 1, Durchführen der Vorgänge 4 bis 5 des hierin zuvor dargelegten Quadrierungsverfahrens, in dem sowohl der Multiplikand als auch der Multiplikator aus dem Zählwerk B stammen, und in dem die aufeinander folgenden Zeichen des Montgomery-Multiplikators aus dem Zählwerk B in das Zählwerk A_1 geladen werden.

10) Wenn das aktuelle Bit des Exponenten E 1 ist, und nur dann, nach Durchführen des Vorgangs 9, Durchführen der Vorgänge 4 und 5 des hierin zuvor dargelegten Multiplikationsverfahrens, in dem der Multiplikand der Inhalt des Zählwerks B und der Multiplikator die Basis A^* ist; und

11) Nach Durchführen der Schritte 8 - 19 aller Bits von E Durchführen einer zusätzlichen Addition nach Montgomery des Zählwerks B mit der ursprünglichen Basis A und dann Speichern des Ergebnisses des letzten Vorgangs $D \text{ ¥ } A^E \text{ mod. } N$ im Zählwerk B, wenn der Exponent ungerade ist; sollte der Exponent gerade sein, Durchführen einer zusätzlichen Multiplikation nach Montgomery von D Mal I: B ¥ ℘(D . 1) ¥ D . I.

Es wird deutlich, daß bei dem Potenzierungsverfahren dieser Erfindung die Notwendigkeit der Berechnung des hierin zuvor erwähnten Parameters T entfällt.

Es wurde weiter festgestellt, und das ist ein weiterer Gegenstand der vorliegenden Erfindung, daß die beschriebene Maschine (in Form eines 512 Bit großen Zählwerks) den Erhalt des Ergebnisses der herkömmlichen Multiplikation von zwei n/2 Bit Zahlen (effektiv jede beliebigen zwei Operanden, die, wenn sie multipliziert werden, kein Ergebnis hervorrufen, das länger als n Bits ist, das heißt, einen Überlauf) ohne den Einsatz zusätzlicher Hardware oder der beschwerlichen Vorgänge erlaubt, die zu seiner Erzielung gemäß den Vorveröffentlichungen erforderlich wären. Dies wird durch Durchführung der modularen Multiplikation besagter Zahlen durch Multiplikationsprozesse erreicht, in denen der Wert des konstanten Koeffizienten N eine n Bit Zahl ist, die aus allen „1s" (fffffff .. fff) besteht, die J_0 bis I entsprechen, und Laden des Multiplikanden in B und Behandeln von A wie in besagtem Multiplikationsvorgang.

Die Vorrichtung zur Durchführung einer derartigen Multiplikation im normalen Zahlenbereich durch das vorgenannte Verfahren kann dieselbe Vorrichtung sein, die Kontrollmittel umfaßt, umfassend eine CPU und einen Multiplikationsschaltkreis, der beinhaltet:

ein n Bit Umschaltzählwerk B für den Multiplikator;

ein n Bit Umschaltzählwerk N for den konstanten Koeffizienten;

ein n Bit Umschaltzählwerk fur den hierin definierten Wert S;

ein k Bit Zahlwerk A_i für den Multiplikanden;

k Bit Zahlwerk Mittel für die hierin definierten Werte J_0 und Y_0

Multiplikationsmittel zur Multiplikation des Inhalts des Zählwerks B mit dem des Zählwerks A_i ;

zusätzliche n-Bit Multiplikationsmittel; und Addier-, Subtraktions-, Multiplex- und Verzögerungsmittel.

Alle Verbindungen zwischen den n Bit Zählwerken und den restlichen Komponenten sind bevorzugt 1 Bit serielle Verbindungen.

KURZE BESCHREIBUNG DER ZEICHNUNGEN

In den Zeichnungen:

ist Figur 1 ein Blockdiagramm eine Geräts gemäß einer erfindungsgemäßen Ausbildungsart;

ist Figur 2 ein Blockdiagramm eines modularen Multiplikations-Schaltkreises gemäß einer erfindungsgemäßen Ausbildungsart;

zeigt Figur 3 den besonderen modularen Multiplikationsschaltkreis gemäß einer erfindungsgemäßen Ausbildungsart;

ist Figur 4 eine schematisches Diagramm, das das zeitliche Verhältnis zwischen den verschiedenen Vorgängen einer Iteration des Multiplikationsvorganges gemäß einer erfindungsgemäßen Ausbildungsart darstellt;

stellt Figur 5 eine serielle / parallele Multiplikationszelle dar;

stellt Figur 6 einen 8 Bit seriellen / parallelen Multiplikator dar,

stellt Figur 7 ein serielles Addierwerk dar;

stellt Figur 8 ein serielles Subtraktionswerk dar,

stellt Figur 9 eine Architektur zur Berechnung des Parameters H dar.

Insbesondere

beschreiben die Figuren verschiedene Ebenen logischer Konzepte, die zum Verständnis der Vorrichtung in ihrer Gesamtheit notwendig sind. In allen Fällen motiviert das Taktsignal den Schaltkreis, und falls es ein 3 Rücksetz-Signal gibt, ist sein Zweck die Initialisierung eines Schaltkreises in den Nullzustand.

Detaillierte Beschreibung bevorzugter Ausbildungsarten

Figur 1 ist ein Blockdiagramm des monolithischen Schaltkreises, in den die Erfindung integriert ist. Der MULT Block enthält die Hardware-Vorrichtung, die die Basis für die Erfindung ist; die Statusmaschine enthält den Controller, der den MULT Schaltkreis antreibt, der ROM Block enthält den gesamten nicht flüchtigen Speicher (ROM und EEPROM), in dem das Programm zur Kontrolle der Smart Cards, die bewährten offentlichen Schlüssel für Dritte und das Programm zum Betrieb des MULT Blocks sowie die Statusmaschine untergebracht sind; der RAM Block enthält den flüchtigen Speicher, der temporäre Operanden wie zum Beispiel zu potenzierende Nachrichten, auf Zugangsberechtigung zu prüfende öffentliche Schlüssel, Daten im Übergang zum MULT Block, usw. speichert; die CPU (Zentrale Prozessoreinheit) kann praktisch jeder bekannte Mikrokontroller sein, der ein 8 Bit oder einen größeren internen Bus hat.

Figur 2 zeigt einen erfindungsgemäßen modularen Multiplikationsschaltkreis in Form eines Blockdiagramms, der zur Durchführung von modularen Quadrierungen und modularen Potenzierungen eingesetzt werden kann. Die Numerale 10, 11 und 12 zeigen drei Zählwerke an, die n Bit lang sind, n = k

m, was jeweils Zählwerke B, S und N bildet, bzw. in die der Multiplikationswert S und der konstante Koeffizient geladen werden. Die vorgenannten Zählwerke werden bevorzugt in zwei Zählwerke n/2 geteilt, bevorzugt unter Einschluß einer k kleinsten Bit Unterteilung für die Zahlwerke N und B. Multiplexer 13, 14 und 15 werden jeweils vor die besagten Zählwerke plaziert und wenn sie in Komponententeile unterteilt werden, wird ein Multiplexer vor jede Unterteilung plaziert. Diese ebenfalls in einem Blockdiagramm gezeigten Zählwerke sind zum seriellen Laden bestimmt, doch wäre es ebenfalls möglich, sie parallel zu laden. 16, 17 und 18 sind drei Zahlwerke, wobei jedes von ihnen k Bits lang ist und jeweils die Werte A_1, J_0 und Y_0 erhalt. Die Zählwerke 16 und 17 sind Umschaltzählwerke mit serieller Ladung / paralleler Ausgabe oder mit serieller und paralleler Ladung / paralleler Ausgabe-Zählwerken. Zählwerk 18 ist bevorzugt ein serielles Umschaltzählwerk in paralleler Ausgabe. Der Inhalt dieser Zählwerke ist zur Verarbeitung durch Multiplikationsmittel 19 und 20 durch Komponenten 21 und 22 bestimmt, welche bevorzugt k Bit Schalter sind. Wenn sie Schalter sind, werden sie von Zählwerken 16, 17 und 18 durch k Bit Busse geladen. Wenn sie Zahlwerke sind, können sie seriell durch I Bit Verbindungen geladen werden. Die Numerale 24, 15, 25', 26, 36, 37 und 38 bezeichnen ebenfalls Multiplexer. Die Multiplikatoren 19 und 20 können A serielle, B parallele Eingaben, serielle Ausgabemultiplikatorenmittel oder jegliche andere serielle / parallele Eingabe- / serielle Ausgabemultiplikationsmittel sein. Der Multiplexer 38 kann den konstanten Koeffizienten N zwingen, alles „len" zur Multiplikation in den normalen Bereichszahlen zu sein.

Die Numerale 27, 28, 29, 30 und 31 bezeichnen 1 Bit volles / halbes Addier- / Subtraktionswerk. 31 bezeichnet ein volles Addier- / Subtraktionswerk. 32, 33 und 34 bezeichnen k Bit k

Taktgeberzyklus-Verzögerungsmittel, die geeignet sind, digital Signale zu verzögern, die aus analogen oder digitalen Komponenten bestehen können, obwohl digitale Komponenten bevorzugt werden. 35 ist ein Entnahmesensor, der ein zwei Bit Schalter-/ Speichermittel ist. Wie dargestellt umfaßt die erfindungsgemäße Vorrichtung, obwohl sie zur Behandlung großer Zahlen bestimmt ist, wie zum Beispiel 512 Bit Zahlen, keine Busse, mit Ausnahme einiger weniger optionaler k Bit Busse, und dies bedeutet eine bedeutende Einsparung an Hardware. Wenn die Zählwerke B, S und n/2 Bit Teile umfassen, kann die erfindungsgemäße Vorrichtung zur Durchführung von Multiplikations- und Potenzierungsvorgängen auf 256 Bit Zahlen eingesetzt werden, was ein bedeutender Vorteil hinsichtlich der Flexibilität des Einsatzes der Vorrichtung ist.

Figur 3 zeigt die logischen Zellen gemäß einer bevorzugten Ausbildungsart der Erfindung, Operanden werden in den A_i Schalter, das Zählwerk J_0, das Zählwerk B und das Zählwerk N über die serielle Verbindung DI gespeist, und Ergebnisse werden über eine serielle Verbindung DO vom Zahlwerk B oder S abgeladen.

Das Signal X ist die Bit Strom-Summenbildung des Produkts von B und A_i und S. (Werte nach S und B haben angenommene Werte, die kleiner sind als N). Das Signal Y_0 ist der k LS Bit Strom des Produkts von J_0 und X. Das Signal Z ist die Summenbildung von X und dem Produkt von Y_0 und N. Die k LS Bits von Z, unter Nichtbeachtung aller Nullen, und nur die n MS Bits werden seriell in S oder B eingespeist.

Der Entnahme-Sensor ist ein logischer Schaltkreis, der feststellt, ob der Wert von $Z/2^K$ is größer ist als N oder nicht.

Die Subtrahierwerke Sub1 und Sub2 subtrahieren den Bit Strom N von den Bit Strömen von B und S, wann immer B oder S großer ist als N.

Ad1 und Ad2 summieren Bit Ströme, um Ströme X und Z zu produzieren.

Die Umschaltzählwerke Delay1 und Delay2 sind notwendig zur Lieferung von Speicher zur Synchronisierung der mathematischen Prozesse.

Es sind keine Taktgeberkontrollen in der Zeichnung enthalten. Es wird vorweggenommen, daß Taktgeber von der Statusmaschine geliefert werden, wann immer Daten entweder von irgendeinem der oben genannten seriell geladenen / abgeladenen logischen Schaltkreisen ausgeht oder in sie eingespeist werden müssen.

Auch andere Kontrollen werden nicht spezifiziert, das heißt, Multiplexeradressen, Schaltertransfersignale, usw., die für Fachleute aus dem in dieser Spezifikation enthaltenem erläuterndem Material naheliegend sein dürften.

Für den Fachmann wird es offensichtlich sein, wie die Vorrichtung der Figur 2 oder der Figur 3 die Vorgänge durchführt, die das erfindungsgemäße Multiplikationsverfahren darstellen. Das zeitliche Verhaltnis besagten Vorgangs wird jedoch weiter in Figur 4 dargestellt. Besagte Figur stellt diagrammartig alle verschiedenen Vorgänge dar, die in effektiven aufeinander folgenden Taktgeberzyklen in einer Ausbildungsart der Erfindung durchgeführt werden, in der n = 512, k = 32 und m = 16 ist. Dies ist eine ziemlich gewöhnliche Situation in der Verschlüsselungskunst. Wenn die Erfindung gemäß der in Figur 3 dargestellten Ausbildungsart durchgeführt wird, kann

dieselbe Vorrichtung ebenfalls zur Durchführung mit n = 256 verwendet werden.

In Figur 4 wird eine Folge verschiedener Vorgänge als eine Funktion der effektiven Taktgeberzyklen dargestellt, die auf der Abszissenachse eingetragen sind. Zu Beginn des Vorgangs und vor irgendeiner der Iterationen, die Teil des erfindungsgemäßen modularen Multiplikationsverfahrens sind, werden die Werte B, N und in die jeweiligen Zählwerke geladen. Das erste Zeichen von A wird ebenfalls in das jeweilige Zählwerk geladen. Sobald eine Iteration beginnt und während k Taktgeberzyklen wird das Umschalten des Inhalts der Register B und S durchgeführt. Die Generation des X-Wertes findet während n+k effektiven Taktgeberzyklen statt, wobei die ersten k Taktgeberzyklen durch Eingabe des Wertes von X_0 besetzt werden. Während der ersten effektiven k Taktgeberzyklen wurde der Wert von Y_0 eingegeben. Während der nächsten effektiven n+k Taktgeberzyklen, wird nun der Wert X, der in den Multiplikator 20 eingeführt worden war, umgeschaltet oder in ein Addierwerk 31 eingeführt, nachdem er durch Verzögerung 34 verzögert worden ist. Der Wert N wird zu drei verschiedenen Zeitphasen verwendet. Erstens, um S und B zu „aktualisieren", zweitens verzögert k effektive Taktgeberzyklen zur Multiplikation mit Y_0 und dann verzögert einen zweiten k effektiven Taktgeberzyklus, um zu erfassen, wie der nächste Wert von S oder B „aktualisiert" werden wird. Während derselben n+k effektiven Taktgeberzyklen wird Z ebenso berechnet wie $Z/2^K$. Der Wert von Ai wird geladen, beginnend mit den ersten k effektiven Taktgeberzyklen und fortführend während des darauffolgenden Teils der Iteration. Der endgültige Wert $Z/2^K$ wird in das Zählwerk S (oder B) während n Taktgeberzyklen nach den ersten 2k effektiven Taktgeberzyklen eingegeben.

Figur 5 zeigt eine Implementierung einer seriellen / parallelen Multiplikationszelle (als Hilfe für diejenigen Fachleute, die mit diesem Fachbereich vertraut sind, denen die Arbeitsweise einer derartigen Konfiguration jedoch vielleicht nicht bekannt ist). Jede dieser Zellen umfaßt einen MPL Block wie in Figur 6 gezeigt. Er implementiert boothsche Multiplikationsalgorithmen für nicht unterzeichnete serielle / parallele Multiplikationen.

Figur 6 zeigt eine Implementierung eines 8 Bit seriellen / parallelen Multiplikators. In den ML1 und ML2 Blöcken der Figur 3 sind die seriellen / parallelen Multiplikatoren k Bits lang. Anzumerken ist, daß die MS Zelle entartet ist. Der parallele 8 Bit Multiplikand wird auf den XI Verbindungen eingegeben und der n Bit lange serielle Multiplikator wird auf dem Y-Verbinder eingegeben (LS Bit zuerst, und eine Folge von k Nullen nach dem MS Bit des Multiplikators). Das Produkt wird auf MO ausgegeben, LS Bit zuerst, MS Bit zuletzt, in dem ein vollständiges Produkt n + k Bits lang ist.

Figur 7 zeigt die seriellen Addierwerke zur Summenbildung von zwei Bit Strömen, die auf A- und B-Eingabeverbindungen erscheinen, und gibt den Summenstrom auf Verbindung S aus. Die LS Bits sind zuerst einzugeben, und der Ausgabestrom für Operanden von m Bits Länge ist m+1 Bits lang. Am Ende des m. effektiven Takts ist die CI-Ausgabe das (m+1). Bit der Zahlenfolge.

Figur 8 zeigt das serielle Subtrahierwerk zur Ausgabe der Differenz zwischen zwei Bit Strömen, die auf den A- und B-Eingabeverbindungen erscheinen und geben den Differentialstrom auf dem D-Verbinder an. Die LS Bits sind zuerst einzugeben, und der Ausgabestrom für Operanden von m Bits Länge ist m Bits lang. Am Ende des m. Bits ist die BI-

Ausgabe das (m+1). Bit der Zahlenfolge und dient als eine Entnahmeanzeige.

Figur 9 zeigt das Hardware-Layout zur Berechnung des Parameters H für einen konstanten Koeffizienten N, das n Bits lang ist. Während dieses Betriebsmodus wird das Zählwerk N für einen n Bit langen konstanten Koeffizienten n + 1 Mal rotiert, auf die Rotierung des Zählwerks S synchronisiert, das durch Subl mit einer Verzögerung des LS Bit (beim ersten Taktzeitgeber in M2_I:1 wird eine LS Null eingefügt) rotiert. Der Entnahmesensor „weiß" am Ende der vollständigen Rotation, ob N in der nachsten Runde vom Strom S subtrahiert wird oder nicht und schaltet den vorhergehenden Subtraktionsmultiplexer dementsprechend für die nächste Runde.

Wie oben angegeben zeigt Figur 1 in Form eines Blockdiagramms eine Vorrichtung zur Durchführung der erfindungsgemäßen Vorgänge. Die Block-KONTROLLE der Vorrichtung umfaßt:

1) Eine vollstandige Zentrale Prozessoreinheit (CPU)

2) Rechner

3) Eine Statusmaschine

Die CPU beinhaltet flüchtigen und nicht flüchtigen Speicher, von dem ein Teil von diesem Multiplikationsvorgang verwendet werden kann. Die CPU kontrolliert den modularen arithmetischen Block im Schaltkreis.

Die CPU:

1) Kommuniziert mit dem Wirtsrechner

2) Lädt Daten auf den Chip und vom Chip herab.

3) Steuert den Schaltkreis zur Durchführung einer Sequenz mathematischer Vorgänge.

4) Ist für andere Verschlüsselungs- und Nichtverschlüsselungs- sowie Datenverarbeitungsprozesse verantwortlich.

Die Rechner generieren die Adresse für die eingebettete Statusmaschine.

Die Statusmaschine decodiert die Adressen und generiert Kontrollsignale an den MULT Block. Diese Kontrollsignale steuern den MULT Block zur Durchführung der entsprechenden Sequenz von Vorgängen, die zur Berechnung der Transformation $\rho(A \cdot B)N$ notwendig sind (wobei A gleich B sein kann).

Figur 3 ist ein Hardware-Blockdiagramm der Hardwarevorrichtung, die die physikalischen Aspekte der Erfindung (MULT) einbettet, und dazu bestimmt ist, bei der Schwerpunktlegung auf mehrere der Architekturkonzepte zu helfen, die durch dieses Patent geschützt werden sollen. Der Block implementiert zusammenfallend die in den Gleichungen (1) bis (5) spezifizierten Sequenzen und ebenso, ohne das synchrone Taktgeben zu verändern, die Umwandlungen von S und B von der begrenzten Kongruenz zur Gleichheit. In diesem Abschnitt nehmen wir vorweg, daß die Konstanten (die Funktionen von N) J_0 und H vorausberechnet worden sind.

Der Schaltkreis führt $\rho(A \cdot B)N$ durch. Unter Verwendung dieser Funktion kann der Schaltkreis genutzt werden, um zu berechnen

1) $B \cdot A \bmod. N$

und

2) $B^2 \bmod. N$,

wobei B immer kleiner sein muß als N.

Implementierung von C = B . A mod. N (A kann gleich B sein)

1) Der Prozessor lädt den Operanden B in das Zählwerk B vor und den Operand N in das Zählwerk N vor.

2) Jedes Mal, wenn der Schaltkreis in MULT mit der Berechnung des nachsten Wertes von S beginnt, signalisiert der Schaltkreis (Zeichen) der CPU, das nächste A_i vorzuladen. Nach der S(m). Iteration befindet sich eine Zahl, die ¥ Kongruenz mit B hat, im Zählwerk B.

3) Block MULT berechnet F = $\wp$(B . H)N, wobei H eine vorausberechnete Konstante in einer den Schritten 1) und 2) identischen Sequenz mit der Ausnahme ist, daß der Prozessor nun die Sequenz von H_i Zeichen vorladen wird (unter Einsatz derselben Sequenz wie der, die beim vorherigen Laden von A_i Zeichen verwendet wurde).

Implementierung von C = B^2 mod. N

1) Unter der Annahme, daß B einen Wert enthält, von dem bekannt ist, daß er ¥ kongruent zu B_i ist, und daß das Zahlwerk N den konstanten Koeffizienten N enthält (wie dies beim Quadrieren allgemein der Fall ist), kann der MULT Block nun die Quadrierung zunachst durch Vorladen des Zählwerks A_i mit B_0, dem LS Zeichen von B_0 vornehmen.

2) Die Berechnung B = $\wp$(B .B)N erfolgt wie der zweite Schritt beim Multiplikationsvorgang, mit der Ausnahme, daß das nachfolgende Laden der B_i Zeichen seriell „fliegend" von den Zählwerken B erfolgt, da die Zählwerke B rotieren.

3) Die Berechnung $\wp$(B - H), falls notwendig, ist identisch mit dem vorgehenden Schritt 3.

Wie dem Fachmann einleuchtet, beanspruchen die Erfinder nicht, daß die seriellen / parallelen (s/p) Multiplikatoren oder irgendeiner der verwendeten herkömmlichen Komponenten *per se* Teil der Erfindung sind. Der folgende Abschnitt wird eingefügt, um die Verwendung von logischen Standardzellen im öffentlichen Bereich zu erläutern, da einige von ihnen nicht im allgemeinen Gebrauch sind. Die hier gezeigte Gatterimplementierung dient nur der Darstellung. Erfahrene Techniker optimieren diese logischen Zellen.

Die Operanden A, B und N sind jeder *n* Bit lang, die aus m Gruppen von k Bit langen Zeichen bestehen, daher n = k . m. In einer Hardware-Implementierung, in der k = 32, kann m entweder 8 oder 16 binäre Bits lang sein.

ML1, ML2

Diese Multiplikatoren führen den Algorithmus nach Booth für nicht signierte Multiplikationen durch, in denen der parallele Operand k Zellen (Bits) lang ist und der seriell geladene Operand von irgendeiner beliebigen geforderten Länge sein.

Jeder serielle / parallel Multiplikator besteht aus k-1 MPL Zellen (Figur 5). Die signifikantesten Zellen, sein MS Bit, besteht nur aus einem UND-Gatter.

Jede MPL Zelle multipliziert die serielle Eingabe Y mit seiner parallelen XI-Bits-Eingabe und summiert dieses Ergebnis mit der seriellen Ausgabe oder vorherigen MPL Einheit und ihren eigenen vorherigen Ausführungsbitzyklen.

Die MPL Zelle ist ein 2 Bit Multiplikations-Addierwerk. Der Block multipliziert das Eingabebit XI und das serielle Eingabebit Y und summiert das Ergebnis mit DI (Daten Ein) und dem Übertrag CI (Übertrag Ein) vom vorherigen Zyklus. Das endgültige Ergebnis ist DO (Daten Aus) und ein CO

(Übertrag Aus) für den nachsten Zyklus. Dieser Übertrag Aus wird in einem Daten Flip-Flop (D F-F) gespeichert.

$$DO = (DI + CI + XI \cdot Y) \bmod 2,$$

und der gespeicherte Übertrag CO wird der CI im nächsten Zyklus sein. Dieser Übertrag ist die boolesche Summe:

$$CO = CI \cdot XI \cdot Y + CI \cdot DI + DI \cdot Y \cdot XI$$

Ad.1, Ad.2

Dies ist ein einfaches 1 Bit volles Addierwerk mit einem D F-F zum Speichern des Übertrags, der im nachsten Taktgeberzyklus (Figur 7) hineingetragen wird.

Die zwei Eingaben A und B werden mit dem Übertrag CI aus dem vorherigen Zyklus summiert, um die Modulo 2 Summe zu generieren, die im D F-F für das Ausgabesignal S gespeichert wird. Beim Zurücksetzen wird das Übertragsbit auf „0" gesetzt.

Sub.1, Sub.2, Sub. 3

Jeder der in Figur 8 beschriebenen Blöcke ist ein volles Subtrahierwerk mit einem Speicher D F-F für die vorherige Entnahme. Dieser Block ist ähnlich dem Ad. 1 Block, mit den Ausnahmen, daß er den B Strom seriell vom A Strom subtrahiert.

Verzög.1, Verzög.2, Verzög.3

Dies sind k Bit Umschaltzählwerke, bestehend aus k 1 Bit verketteten Speichervorrichtungen. Sie werden zur Synchronisierung der verschiedenen Operanden in der mathematischen Sequenz eingesetzt. Dies wird deutlich, wenn der Schaltkreis erläutert wird.

A+, J_0, Y_0

Diese Blöcke sind k Bit lange serielle Ein- / parallele Aus-Umschaltzählwerke, in die k Eingabebits seriell eintreten. Nach k effektiven Taktgeberzyklen erscheinen diese k Bits in der Ausgabe parallel.

In Figur 2 sind die dünnen Linien serielle Ein-Bit-Leiter und die dicken Linien stellen k Bit parallele Leiter dar.

M4_I:x M3_I:x M2_I:x

Dies sind Ein-Bit-Ausgabe-Multiplexer - M4_I:x, die 1 von 4 Eingaben ausgeben - M3_I:x, die 1 von 3 Eingaben ausgeben und M3_I:S, die 1 von 2 Eingaben ausgeben, x stellt den expliziten Index einer spezifischen Komponente dar.

B(0:k-1), B(k:n1-1), B(n1:n2), S(0:n-1), S(n1:n2), N(0:k-1), N(k:n1-1), N(n1:n2)

Dies sind Umschaltzählwerke. Die Größe und der Platz in der Sequenz eines längeren Zählwerks werden durch die Zahlen in Klammern bezeichnet, zum Beispiel ist X(s:t) ist ein t - s + 1 Bit langes Umschaltzählwerk, s ist der Index für das erste Bit von X(s:t) und t ist der Index des letzten Bit des Zählwerks X(s:t). Zum Beispiel besteht B(0:511) aus den drei kürzeren kaskadierten Zählwerken: B(0:31), B(32:255) und B(256:511).

n1 ist im allgemeinen gleich n/2, zum Beispiel muß 256 . n1 ein Vielfaches von k sein.

n2 ist gleich n-1.

k ist die Länge des Maschinenzeichen, das heißt, die Größe der seriellen / parallelen Multiplikatoren. Daher werden die folgenden Werte in der ersten Implementierung vorweggenommen: n1 = 256, n2 = 511, n = 512 und k = 32.

Schalter1, Schalter2

Diese zwei Schalter sind K Bit Zählwerke. Sie werden zur Sperrung der parallelen Daten in dem Multiplikator zur Ermöglichung von parallelen Einzeltakttransaktionen in Multiplikationssequenzen eingesetzt.

MULT Blockvorgang - Multiplikationen und Potenzierungen im φ-Bereich

Zur Erleichterung der Erläuterung haben wir uns entschlossen, nur solche Taktgeberzyklen zu bezeichnen, die tatsächlich Daten in Zählwerken bewegen; diese „bewegenden" Zyklen definieren wir als „effektive Taktgeberzyklen".

φ(A . B)N Multiplikation

Stufe 1: Erstes Laden

Folgende Zählwerke werden durch DI geladen:

1) J_0 in das Zählwerk J_0 (von CPU vorausberechnet)

2) B in das Zählwerk B

3) N in das Zählwerk N

4) Das erste Zeichen von A, A_0 in das Zählwerk A_2

Gleichzeitig zu Schritt 2 wird Zählwerk S mit Nullen geladen.

Nach dem Laden dieser fünf Zählwerke werden die zwei parallelen seriellen nicht signierten Multiplikatoren ML1, ML2, die seriellen Addierwerke AD1 und AD2 und die seriellen Subtrahierwerke Sub1, Sub2 und Sub3 zurückgesetzt.

Stufe 2: Ausführen der Iteration B . A_0

Die in das Zählwerk A_1 geladenen Daten A_0 werden in Schalter1 geladen. Zählwerk B wird zyklisch nach rechts umgeschaltet.

Bei Initialisierung des Prozesses steht das Entnahme2 Kontrollsignal auf „0"; daher geht der Inhalt von B einfach unverändert durch das Subtrahierwerk Sub1 und wird in ML1 mit A_0 multipliziert. Die Ausgabe des Zählwerks B wird unverändert in die Eingabe des Zählwerks zurückgespeist.

Das Ergebnis dieser Multiplikation wird seriell in Ad1 zum Inhalt des Zählwerks S addiert, das bei dieser ersten Iteration überall Null ist. Dieser Vorgang generiert X wie hierin zuvor beschrieben.

Während diese Vorgange voranschreiten, lädt die CPU das nächste Zeichen von A, A_1 in Schalter1.

J_0 aus dem Zählwerk J_0 wird in Schalter2 geladen. X wird seriell in ML2 zur Multiplikation mit J_0 eingegeben. Somit ist der Inhalt des Zählwerks Y_0 nach k effektiven Taktgebern die k kleinsten Bits des Produkts $X_0 . J_0$.

Dann, nach diesen ersten k effektiven Taktgebern, wird ML3 zurückgesetzt; der serielle Eingabemultiplexer M3_1:4 wird vom Strom X auf den Strom N umgeschaltet; die Daten im Zählwerk Y_0 werden anstelle von J_0 parallel in Schalter2 geladen; und die Ausgabe wird auf den Strom $Y_0 . N$ umgeschaltet. Für die nächsten n+k effektiven Taktgeberzyklen wird das serielle Ausgabeergebnis der ML2-Multiplikation $Y_0 . N$ sein. X, das durch k effektive Taktgeberzyklen verzogert wurde, wird nun in Ad2 zum Produktstrom von ML2 summiert; dies generiert $Z = X + Y_0 . N$; eine Zahl, in der die k kleinsten Bits gleich Null sind.

Da die ersten k Bits von Ad2 alle Null sind, bleiben sie unbeachtet und die nächsten n Bits werden seriell an das Zählwerk S zurückgesendet. Diese endgültige Menge kann größer als oder gleich N sein (in diesem Fall muß sie um N reduziert werden), das heißt, $S(1) \neq S(1) \bmod N$.

Um herauszufinden, of $S \geq N$, wird N seriell von diesem n Bit langen ($2/2^k$) Strom in Sub3 subtrahiert. Jedoch wird nur das n. Entnahmebit in diesem Entnahme-Speicher Flip-Flop gespeichert.

Wenn dieses Entnahmebit „0" ist oder das endgültige Übertragsbit CO des Addierwerks Ad2 „1" ist, dann ist der neue Wert in S größer als N.

Am Ende dieser ersten Iteration gibt es einen Wert im Zählwerk S, der die ¥ begrenzte Kongruenz von S(1) mod. N ist; die Zählwerke J_0, B und N halten die ursprünglichen Werte zurück, mit denen sie geladen wurden; und das Vorladezählwerk A_i beinhaltet A_1.

Stufe 3: Nachfolgende Iterationen B . A_i

Das nächste Zeichen von A, A_i wird in den Schalter1, die parallele Eingabe von ML1, geladen.

Während der nächsten und nachfolgenden Iterationen B A_i ist der Inhalt von S am Ende jeder Iteration ¥ zu S(1) mod. N. Wenn S(i) : N, dann ist N von S(i) in Sub.2 zu subtrahieren.

Bei Beginn jeder Iteration wird das nächste Zeichen von A, A_i durch die CPU in das Vorladezählwerk A_i geladen.

<u>℘(B . B)N Quadrierungsvorgänge</u>

Der erste Vorgang bei einer normalen Potenzierung ist ein Potenzierungsvorgang, der wie eine normale Multiplikation mit dem in das Zahlwerk B geladenen Multiplikator A und dem in das in k Bit Zunahme in das Zählwerk A_i geladene Multiplikand wie im vorherigen Abschnitt beschrieben durchgeführt wird. Nachfolgende Quadrierungen werden auf Operanden (Multiplikatoren und Multiplikanden) durchgeführt, deren begrenzte Kongruenz im Zahlwerk B liegt.

Während solcher ρ(B .B)N Quadrierungen werden J_0, S, B und N von Beginn an bereits von einer vorherigen Multiplikation oder Quadrierung geladen und bleiben unverändert; bei jeder Iteration jedoch muß das Zählwerk A_1 mit einem neuen Zeichen geladen werden, das von einem k Bit Zeichen abgeleitet wird, das sich im Zählwerk B befindet.

Für diese nachfolgenden Quadrierungen wird das Zahlwerk A_1 „fliegend" vom Strom B vorgeladen. Sobald die CPU die Steuerquadrierung gegeben hat, hat sie während der nachfolgenden Quadrierungsvorgänge B . B_1 keine Aufgabe auszuführen. Die B_{15}, die geladen werden, sind Segmente von B, die durch Subi geflossen sind (B_1-Segmente von B, die bereits kleiner sind als N).

Stufe 1: Iteration B . B_0

Ursprünglich befindet sich das letzte ¥ von S aus der vorherigen Berechnung im Zählwerk B.

Die k LS Bits der Zählwerke B und N werden zyklisch nach rechts umgeschaltet, damit werden die Zählwerke B und N nach k effektiven Taktgebern in ihre ursprünglichen Zustände zurückgespeichert. Der Wert im Zählwerk B ist entweder der eigentliche Wert B oder der Wert B-N, der für die nächste ρ-Multiplikation zu verwenden ist. So ist das Zählwerk A_1 für die erste Runde entweder mit B_0 vorzuladen, das sich im Zahlwerk B befindet, oder den k LS Bits von B-N.

Der Zweck dieser ersten k Bit Rotation ist es, die ersten k Bits der Vorladung für das Zahlwerk A_1 durch Subl strömen zu konnen. Unverzüglich nachdem es seriell geladen wurde, wird A_1 in Schalterl abgeladen, und das Vorladezahlwerk A_1 ist frei, um mit B_1, dem zweiten Zeichen von B, geladen zu werden.

Während dieses und nachfolgender Vorgänge ist die Ausgabefolge von Sub1 positiv und immer kleiner als N, da das Signal Entnahme2 gesetzt oder zurückgesetzt wird.

Nun, da alle Werte in die Zählwerke geladen wurden, führt diese erste Multiplikation ähnlich wie die Iteration B . A_0 vor, gemäß Beschreibung im vorherigen Abschnitt, mit der Ausnahme, daß B rotiert, wie noch erläutert wird. B1 wird in das Zahlwerk A_1 geladen (es sei daran erinnert, daß die CPU in einer Multiplikation das Zählwerk A_i lädt).

Als das zweite k Bit Zeichen taucht B_1 während dieses ersten Verfahrens B . B_0 B_1 aus dem Strom B auf. das Segment B_1 seriell „fliegend" in das Vorlade-Zählwerk A_1 umgeschaltet, das für den nächsten Quadrierungsvorgang in Vorbereitung ist, das heißt die Iteration B . B_1.

Stufe 2: Iteration B . B_1

Der in das Zählwerk A_1 geladene Wert B_1 wird an seine Ausgabe Schalter1 transferiert. Während der nächsten n + 2k (zum Beispiel n + 64) effektiven Taktgeberzyklen wird das Multiplikationsverfahren auf B . B_1 wie oben beschrieben durchgeführt.

Wie zuvor bestimmen die Signale Entnahme1 und Entnahme2, ob N von den Strömen zu subtrahieren ist, die von den Zählwerken B und S ausgehen. Wenn die Zahl im Zählwerk S größer als oder gleich N ist, dann wird Entnahme1 gesetzt und N wird mit dem Subtrahierwerk Sub.2 von S subtrahiert. N wird, falls notwendig, für die Dauer einer vollständigen m-Iterations-Multiplikationsschleife von B subtrahiert. Eine derartige Bedingung würde mit Entnahme2 am Ende der vorherigen Multiplikation oder Quadrierung erfaßt werden.

Die zwei Flip-Flops, Entnahme1 und Entnahme2 enthalten die endgültigen Werte der konditionierten Entnahme Aus von Sub3.

Entnahme1 wird nach jeder Iteration von S gesetzt oder zurückgesetzt. Entnahme2 wird nach der letzten Iteration S(m) gesetzt oder zurückgesetzt, während B mit S(m) geladen wird. Die konditionierte Entnahme Aus ist das Signal, das anzeigt, ob an S(1) größer ist als N.

Während der Sequenz $B \cdot B_1$ wird das Zeichen B_2 „fliegend" in das Vorladezählwerk A, geladen, da das Zeichen B_0 im Subtrahierwerk Sub. 1 existiert.

Stufe 3: Nachfolgende Multiplikationsiterationen $B \cdot B_1$

Die verbleibenden Iterationen m-2 werden durchgeführt: während jeder einzelnen wird das Zählwerk A_1 mit dem Wert des Zeichens B_1 geladen, da es in Sub.1 in Vorbereitung für die nächste Schleife existiert.

Das endgültige Ergebnis, eine begrenzte Kongruenz, befindet sich sowohl im Zählwerk S als auch B. Diese Daten werden, falls nötig, bei Sub.1 berichtigt, wenn sie seriell durch DO ausgegeben werden.

Vorgang MULT Block - Berechnung des Parameters H

Zur Berechnung von H wird die Maschine neu konfiguriert, um die Zählwerke S und N wie in Figur 9 einzusetzen. Wir demonstrieren den Vorgang des Operators unter Einsatz des bereits oben verwendeten numerischen Beispiels. Diese Konfiguration führt in n+1 Runden eine Berechnung H durch. Bei jeder Runde werden sowohl S als auch N rotiert, wobei jede Rotation n effektive Taktgeber sind. Bei jeder Runde zirkuliert N und kehrt unverändert zurück. Am Ende der 1. Runde enthalten S und das Signal „nächste Subtraktion" das Äquivalent einer begrenzten Y Kongruenz von S(1).

Die Ausgangsbedingungen - 1. Runde

Zu Beginn der ersten Runde wird der konstante Koeffizient N in das Zählwerk N geladen, und das Entnahmesensorzeichen

wird zurückgesetzt, was bedeutet, daß die erste Versuchssubtraktion erfolgreich sein wird; der Ausgabe-Flip-Flop von Sub.1 wird auf Null zurückgesetzt. Für Runde 1 wissen wir, daß das MS (n.) Bit des Versuchsdividenden Eins ist. Dieses Bit wird durch Deduktion im Flip-Flop „nachste Subtraktion" (kein Platz in S) gespeichert. Die „nächste Subtraktion" steuert die Subtraktion S-N in Runde 1.

Unter Verwendung des oben beschriebenen numerischen Beispiels n = 4 Bit demonstrieren wir den Berechnungsmodus H - Ausgangsbedingung.

Gespeichert im Zeichen nächste Subtraktion des Entnahmesensors

Zu Beginn wissen wir, daß das MS Bit des Dividenden „1" ist.

Daher, weil wir wissen, daß es keine Entnahme geben könnte, setzen wir das Zeichen nächste Subtraktion wieder auf Null.

⇓

Das Signal Entnahmesensor - nachste Subtraktion

ist eine Null, daher wird M2_1;3 in der ersten Runde N in Sub1 einspeisen. Die Differenz wird S=N mit einer vorstehenden Null sein, oder, genauer

N = 1011, n = 4

Siehe Figur 7

S(0) Die Inhalte des Zählwerks S

⇓

(0) 0000 (0000) ⇐ „Virtuelle Nullen"

(Diese „virtuellen" LS Nullen werden von einer Versuchssubtraktion nicht beeinträchtigt. Bei jeder Runde wird es eine Null weniger im „virtuellen

ausgedrückt, 2 . (S-N). Nullenzähler" geben)

Im ersten Taktgeberzyklus wir die Null vom Zurücksetzungs-Subl.- Ausgabe Flip-Flop in S's MS Zelle S eingespeist, ebenso wie das LS Bit vom S in Sub.1 eingespeist wird.

(Das LS Bit von S ist immer eine Null, die aus dem „virtuellen" LS Null-Zähler stammt.)

Während der ersten n-1 Taktgeberzyklen werden die LS n-1 Bits von Diff in S eingespeist.

N wird zurück in seine MS Bit Zelle rotiert.

Der Strom BO (Entnahme Aus) ist gleich mit den Serien der Entnahmen, die aus dem Strom

Diff. mod. 2^n - N

resultieren, allerdings wird nur die letzte Entnahme abgefragt und kann relevant sein.

Beim n. effektiven Taktgeberzyklus wird

„nächste Subtraktion" ein Zeichen für eine Subtraktion für die nächste Runde hervorrufen, wenn das MS Bit von Diff. „1" ist ODER wenn BO = „0".

In der ersten Runde wird N von 2" subtrahiert und n Bits des mit 2 multiplizierten Ergebnisses (eine LS Null-Einfügung) in das Zählwerk S zurückgeführt, MIT AUSNAHME des MS Bit, das „durch Deduktion" im Zählwerk Entnahme-Sensor __ nächste Subtraktion gespeichert wird.

Am Ende der ersten Runde rotieren:

S(1) = 1010, nächste Subtraktion = 1 (80 = 1), und wir wissen, daß es in der nächsten Runde keine Subtraktion von S-N in Sub.1 geben wird.

Berechnung des Parameters H . 2. Runde

Gespeichert im Zeichen nächste Subtraktion des Entnahmesensors

Zu Beginn wissen wir, daß die Subtraktion der zweiten Runde nicht erfolgreich wäre, da BO = „1", „erfaßt" als Sub.2.

N = 1011, n = 4

S(1) Die Inhalte des Zählwerks S nach der ersten Runde

(1) 1010 (000) ⇐ „3 Virtuelle Nullen" übrig

Das Signal Entnahmesensor - nächste Subtraktion ist Eins, daher wird M2_1;3 in dieser Runde Nullen in Sub1 einspeisen. Diff. = 2 . S

ES GAB KEINE SUBTRAKTION.

(Das LS Bit von S ist wieder eine Null, die aus dem „virtuellen" LS Null-Zähler stammt.)

Für die nachfolgenden n-1 Taktgeberzyklen werden die LS n-1 Bits der Diff. = 2 . S in das

Zählwerk S eingespeist.

N wird zurück in seine MS Bit Zelle rotiert.

Da das MS Bit der Diff. eine „1" ist, wissen wir, daß wir in der nächsten Runde S - N subtrahieren müssen.

Die abgefragte B0 ist irrelevant.

Diff. = 1 0100 und S(2) = 0100, nächste Subtraktion = 0, und wir wissen, daß es in der nächsten Runde eine Subtraktion von S - N in Sub.1 geben wird.

Berechnung des Parameters H . 3. Runde

Gespeichert im Zeichen nachste Subtraktion des Entnahmesensors

Zu Beginn wissen wir, daß die dritte Subtraktionsrunde erfolgreich sein wird, da das MS Bit von Diff. „1" war.

⇓ N = 1011, n = 4

S(2) Die Inhalte des Zählwerks S nach der zweiten Runde

⇓

⇓ (0) 0100 (00) ⇐ „2 Virtuelle Nullen" übrig

Das Signal Entnahmesensor - nächste Subtraktion ist eine Null, N wird von Diff. subtrahiert.

Für die nachsten n-1 Taktgeberzyklen werden die LS n-1 Bits von Diff. = 2(S-N) zuruck in das Zählwerk S eingespeist.

Da das MS Bit von Diff. in Sub. 1 „1" ist, müssen wir in der nächsten Runde S-N subtrahieren,

Diff. = 1 0010 und S(3) = 0010, nächste Subtraktion = 0, und wir wissen, daß es in der nachsten Runde eine Subtraktion von S-N in Sub.1 geben wird.

Berechnung des Parameters H . 4. Runde

Gespeichert im Zeichen nächste Subtraktion des Entnahmesensors

Zu Beginn wissen wir, daß die vierte Subtraktionsrunde erfolgreich sein wird, da das MS Bit von Diff. „1" war.

⇓ N = 1011, n = 4

S(3) Die Inhalte des Zahlwerks S nach der dritten Runde

⇓ ⇓

— — (0) 0010 (0) ⇐ „1 Virtuelle Null" übrig

⇓

Das Signal Entnahmesensor - nächste Subtraktion ist eine Null, N wird von Diff. subtrahiert.

Da es keine Entnahme BO = „0" gab, werden wir in der nachsten Runde S-N subtrahieren.

Diff. = 0 1110 und S(4) = 1110, nächste Subtraktion = 0, und wir wissen, daß es in der nachsten Runde eine Subtraktion von S-N in Sub. 1 geben wird.

Berechnung des Parameters H . n+1. (5.) Runde

Gespeichert im Zeichen nächste Subtraktion des Entnahmesensors

Zu Beginn wissen wir, daß die vierte Subtraktionsrunde erfolgreich sein wird, da das MS Bit von Diff. „1" war.

N = 1011, n = 4

.S(4) Die Inhalte des Zählwerks S nach der vierten Runde

— — (0) 1110 ( ) ⇐ „Keine Virtuelle Null" übrig

Das Signal Entnahmesensor - nachste Subtraktion

Letzte Runde

ist eine Null, N wird von Diff. subtrahiert.

Diff. = 0 0011 und S(5) = 0011, ist der Rest - der der Wert von H ist.

0601907

ANSPRÜCHE

1. Mikroelektronisches Gerät zur Durchführung der modularen Multiplikation eines Multiplikators durch einen Multiplikanden, wobei das Gerät Addierwerke (30) und Zählwerke (10, 11, 12) umfaßt, dadurch gekennzeichnet, daß das Gerat umfaßt:

erste (10), zweite (11) und dritte (12) Hauptumschalt- und getaktete serielle-Ein-, serielle Aus-Zahlwerke, die jeweils geeignet sind, den Multiplikator, ein Teilergebnis und einen konstanten Koeffizienten zu speichern;

ein erstes serielles / paralleles Multiplex-Multiplikationswerk (19), in dem der Multiplikand untergebracht ist und das geeignet ist, für jede einer Vielzahl von Teilen des Multiplikanden seinerseits den Multiplikator aus dem ersten Zählwerk (10) zu empfangen, den Multiplikator durch einen laufenden Teil des Multiplikanden (21) zu multiplizieren und eine Ausgabe zu generieren, umfassend ein Produkt besagter Multiplikation;

ein Subtrahierwerk (28) zum Subtrahieren des konstanten Koeffizienten von den Inhalten des zweiten Zählwerks (11), zur Produktion einer begrenzten Kongruenz davon, in der, nachdem die Vielzahl der Teile des Multiplikanden durch das erste Multiplikationswerk (19) verarbeitet worden ist, besagtes Teilergebnis eine begrenzte Kongruenz eines Durchführungsergebnisses besagter modularer Multiplikation besagten Multiplikationswerkes durch besagten Multiplikanden bildet;

ein serielles Addierwerk (30), das auf die Ausgabe des ersten Multiplikationswerkes (19) einwirkt und eine begrenzte Kongruenz des Teilergebnisses, das sich in dem zweiten Zählwerk (11) befindet und geeignet ist, eine Ausgabe zu liefern;

ein zweites serielles / paralleles Multiplex-Multiplikationswerk (20), das in einer ersten Phase die Ausgabe des seriellen Addierwerkes (30) und eine Montgomery-Konstante empfängt und in einer zweiten Phase den konstanten

Koeffizienten aus dem dritten Zählwerk (12) empfängt und in der ersten Phase geeignet ist, ein Produkt der ersten Phase der Montgomery-Konstante durch einen Teil der Ausgabe des seriellen Addierwerks (30) zu berechnen und in der zweiten Phase den konstanten Koeffizienten mit dem Produkt der ersten Phase zu multiplizieren und dadurch eine Ausgabe der zweiten Phase zu generieren, die, wenn sie mit der Ausgabe des seriellen Zählwerks (31) kombiniert wird, besagtes Teilergebnis generiert;

Schaltelemente (23) zur Lieferung von Differentialeingaben an wenigstens besagtes zweites Multiplikationswerk (20), jeweils in der ersten und zweiten Phase;

ein zweites Subtrahierwerk (27) zum Subtrahieren des konstanten Koeffizienten von den Inhalten des Zählwerks (10), zum Produzieren der Inhalte des vom konstanten Koeffizienten reduzierten Zählwerks (10), in dem besagtes erstes Multiplikationswerk (19) ein erstes serielles / paralleles Multiplikationswerk (19) umfaßt, das besagte Inhalte des vom konstanten Koeffizienten reduzierten Zahlwerks (10) seriell empfängt und den Multiplikanden parallel empfängt;

eine Entnahme-Sensor-Vorrichtung (35), geeignet zum Empfang der Ausgabe des zweiten Addierwerks (31) und zur Bestimmung, ob die Ausgabe des zweiten Addierwerks (31) größer als oder gleich ist wie der konstante Koeffizient;

in dem die Längen der ersten (19) und zweiten (20) Multiplikationswerke beide k sind, wobei das Gerät ebenfalls ein zweites Addierwerk (31) umfaßt, das geeignet ist, die Ausgabe des seriellen Addierwerks (30) mit einer Verzögerung von k effektiven Taktgeberzyklen zu empfangen und die Ausgabe der zweiten Phase des zweiten Multiplikationswerkes zu empfangen und diese Ausgaben zu addieren und dadurch eine Ausgabe des zweiten Addierwerks zu generieren, in der die k kleinsten Bits Null sind, und besagte Ausgabe des zweiten Addierwerks in eine erste (30) ausgewählte des Zählwerks (10) oder des Zählwerks (11) einzuspeisen; und eine k-Bit Verzögerungseinheit (34) zwischen dem ersten (30) und dem zweiten (31) Zählwerk, die geeignet ist, die Verzögerung von k effektiven Taktgeberzyklen zu liefern.

2. Gerät gemäß Anspruch 1, dadurch gekennzeichnet, daß besagtes erstes Multiplikationswerk (19) einen ersten Eingabeschalter beinhaltet, in dem der Multiplikand untergebracht ist.

3. Gerät gemäß Anspruch 1 und 2, dadurch gekennzeichnet, daß das zweite Multiplikationswerk (20) einen zweiten Eingabeschalter (22) beinhaltet, der den Multiplikanden empfängt.

4. Gerät gemäß Anspruch 1 bis 3, dadurch gekennzeichnet, daß er kein anderes Multiplikationswerk einsetzt als besagte erstes und zweites Multiplikationswerk (19, 20).

5. Gerät gemäß Anspruch 1 bis 4, dadurch gekennzeichnet, daß eine Montgomery-Konstante JO im zweiten Multiplikationswerk (20) in der ersten Phase untergebracht ist und die Ausgabe der ersten Phase des zweiten Multiplikationswerkes im zweiten Multiplikationswerk (20) in der zweiten Phase untergebracht ist.

6. Gerät gemäß Anspruch 5, dadurch gekennzeichnet, daß es einen Bereich (23) von 2-bis-1 Multiplexern umfaßt, die geeignet sind, die Montgomery-Konstante in das zweite besagte Multiplikationswerk (20) in der ersten Phase einzuspeisen und die Ausgabe der ersten Phase des zweiten Multiplikationswerkes (20) in das zweite Multiplikationswerk (20) in der zweiten Phase einzuspeisen.

7. Gerät gemäß Anspruch 6, dadurch gekennzeichnet, daß es ein serielles / paralleles Zählwerk umfaßt, das die Ausgabe des zweiten Multiplikationswerkes (20) in der ersten Phase empfängt und die Ausgabe parallel in besagtes zweites Multiplikationswerk (20) über besagten Multiplexerbereich (23) in der zweiten Phase einspeist.

8. Gerät gemäß Anspruch 7, dadurch gekennzeichnet, daß besagter Multiplex-Bereich (23) k 2-bis-1 Multiplexer umfaßt und in dem besagtes serielles / paralleles Zählwerk die Länge k hat.

9. Gerät gemäß Anspruch 1, dadurch gekennzeichnet, daß das erste serielle Subtrahierwerk (27), das die Inhalte des Zählwerks (10) empfängt und daraus den konstanten Koeffizienten subtrahiert, einen modular reduzierten Multiplikator berechnet, wenn die Ausgabe des zweiten Addierwerks (31) größer als oder gleich wie der konstante Koeffizient ist, und den modular reduzierten Multiplikator in besagtes erstes Multiplikationswerk (19) einspeist.

10. Gerät gemäß Anspruch 9, dadurch gekennzeichnet, daß es eine Vergleichseinrichtung umfaßt, die bestimmt, ob die Ausgabe des zweiten Addierwerks (31) größer als oder gleich dem konstanten Koeffizienten ist, in dem die Vergleichseinrichtung operativ mit dem Subtrahierwerk (28) verbunden ist, so daß das Subtrahieren des konstanten Koeffizienten von den Inhalten des Zählwerks (11) kontrolliert wird.

11. Gerät gemäß Anspruch 10, dadurch gekennzeichnet, daß es eine Vergleichseinrichtung umfaßt, die bestimmt, ob die Ausgabe des zweiten Addierwerks (31) größer als oder gleich dem konstanten Koeffizienten ist, in dem die Vergleichseinrichtung mit dem Subtrahierwerk (27) operativ verbunden ist, so daß das Subtrahieren des konstanten Koeffizienten von den Inhalten des Zählwerks (10) kontrolliert wird.

12. Gerät gemäß Anspruch 1, dadurch gekennzeichnet, daß das Subtrahierwerk (28), das die Inhalte des Zählwerks (11) empfängt und davon den konstanten Koeffizienten subtrahiert, einen modular reduzierten Multiplikator berechnet, wenn die Ausgabe des Addierwerks (30) größer als oder gleich dem konstanten Koeffizienten ist, und die modular reduzierte Ausgabe des Zählwerks (11) in besagtes Multiplikationswerk (20) einspeist.

13. Gerät gemäß Anspruch 1, dadurch gekennzeichnet, daß besagtes zweites Addierwerk (31) ein serielles Addierwerk umfaßt.

14. Gerät gemäß Anspruch 1, dadurch gekennzeichnet, daß besagtes erstes und zweites Multiplikationswerk (19, 20) jedes eine Länge k hat und in denen die Dauer der ersten Phase k effektive Taktgeberzyklen sind.

15. Gerät gemäß Anspruch 1, dadurch gekennzeichnet, daß das serielle Addierwerk (30) auf die Ausgabe des ersten Multiplikationswerkes (19) und auf die modular reduzierten Inhalte des Zählwerks (11) einwirkt.

16. Gerät gemaß Anspruch 1, dadurch gekennzeichnet, daß besagter Hauptschalter und besagte getaktete Zählwerke (10, 11, 12) unterteilt sind.

17. Gerät gemäß Anspruch 1, dadurch gekennzeichnet, daß die Ausgabe der zweiten Phase eine serielle Ausgabe umfaßt.

18. Ein Verfahren zum Einsatz mikroelektronischer Potenzierungsgerate zur Durchführung modularer Quadrierung und modularer Multiplikation eines Multiplikators durch einen Multiplikanden, dadurch gekennzeichnet, daß das Verfahren nachfolgende Schritte umfaßt:

- Speichern des Multiplikators, eines Teilergebnisses und eines konstanten Koeffizienten in ersten (B), zweiten (S) und dritten (N) unterteilten Hauptschalt- und getakteten jeweils seriellen-Ein seriellen-Aus Zählwerken (10, 11 und 12);

für jedes einer Vielzahl Teile des Multiplikanden (21), der in einem ersten seriellen-parallelen Multiplikationswerk (19) untergebracht ist, der den Multiplikator aus dem Zählwerk (10) in besagtem seriellen-parallelen Multiplikationswerk (19) empfängt und seinerseits den Multiplikator mit einem laufenden Teil des Multiplikanden multipliziert und eine Ausgabe generiert, umfassend ein Produkt besagter Multiplikation;

in dem, nachdem die Vielzahl Teile des Multiplikanden von dem ersten Zählwerk (19) verarbeitet worden ist, besagtes Teilergebnis eine begrenzte Kongruenz eines Ergebnisses der Durchführung besagter modularer Multiplikation besagten Multiplikators durch besagten Multiplikanden bildet;

- Addieren der Ausgabe des ersten seriellen Addierwerks (30) des ersten Multiplikationswerkes (19) mit einer begrenzten Kongruenz des sich in dem Zählwerk (11) befindenden und eine Ausgabe liefernden Teilergebnisses;

- erst in einem zweiten seriellen-parallelen Multiplex-Multiplikationswerk (20), das in einer ersten Phase die Ausgabe des ersten seriellen Zählwerks und eine Montgomery-Konstante empfängt und in einer zweiten Phase den konstanten Koeffizienten aus dem Zählwerk (12) empfängt und in der ersten Phase ein Produkt der ersten Phase der Montgomery-Konstante durch einen Teil der Ausgabe des ersten seriellen Zahlwerks (30) berechnet und in der zweiten Phase den konstanten Koeffizienten mit dem Produkt der ersten Phase multipliziert, um dadurch eine Ausgabe der zweiten Phase zu generieren;

- in einem zweiten seriellen Addierwerk (31) die Ausgabe der zweiten Phase mit der Ausgabe des ersten seriellen Zählwerks (30) kombinieren, um dadurch besagtes Teilergebnis zu generieren;

- in den Subtrahierwerken (27, 28) den konstanten Koeffizienten von den Inhalten der Zählwerke (10, 11) subtrahieren, um eine begrenzte Kongruenz davon zu produzieren;

- aktivieren durch einen Entnahme-Sensor (35), der geeignet ist, die Subtrahierwerke (27, 28) zu aktivieren;

- Differentialeingaben an wenigstens besagte zweite Multiplikationswerke (20) jeweils in der ersten und zweiten Phase liefern;

- Durch Verzögerungszahlwerke (32, 33, 34) wenigstens besagte erste und zweite Phase synchronisieren; und

- besagtes Gerät durch Durchführung von wenigstens einer modularen Multiplikationsoperation oder einer modularen Quadrierungsoperation einsetzen.

Smart Card
Interface
ISO 7816
RAM
ROM
CPU
Rechner
Status
Machine
Speicher Bus
Kontrollesbus
MULT
Modulare Multiplikation
Quadrierungsschaltkreis

Fig. 1

Fig. 2

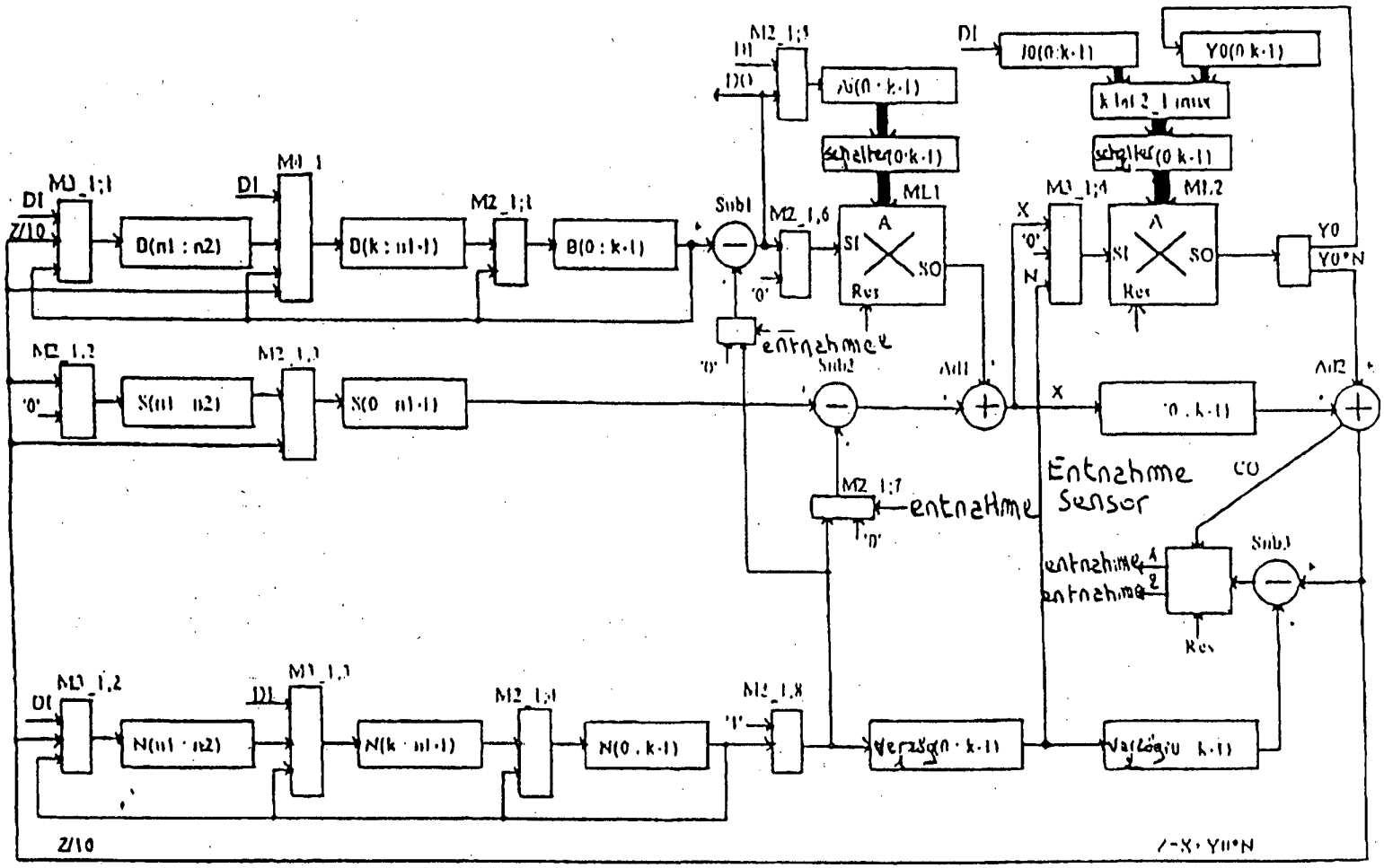
DI
Z/10
D(n1 : n2)
D(k : n1-1)
B(0 : k-1)
Sub1
M2_1;6
M2_1;9
M1_1
M1_2
M3_1;4
SI
SO
Res
entnahme
Sub2
Add1
Add2
S(n1 : n2)
S(0 : n1-1)
M2_1;7
Entnahme
Sensor
CO
Sub3
M3_1;2
M3_1;3
M2_1;5
M2_1;8
N(n1 : n2)
N(k : n1-1)
N(0 : k-1)
Z/10

Fig. 3

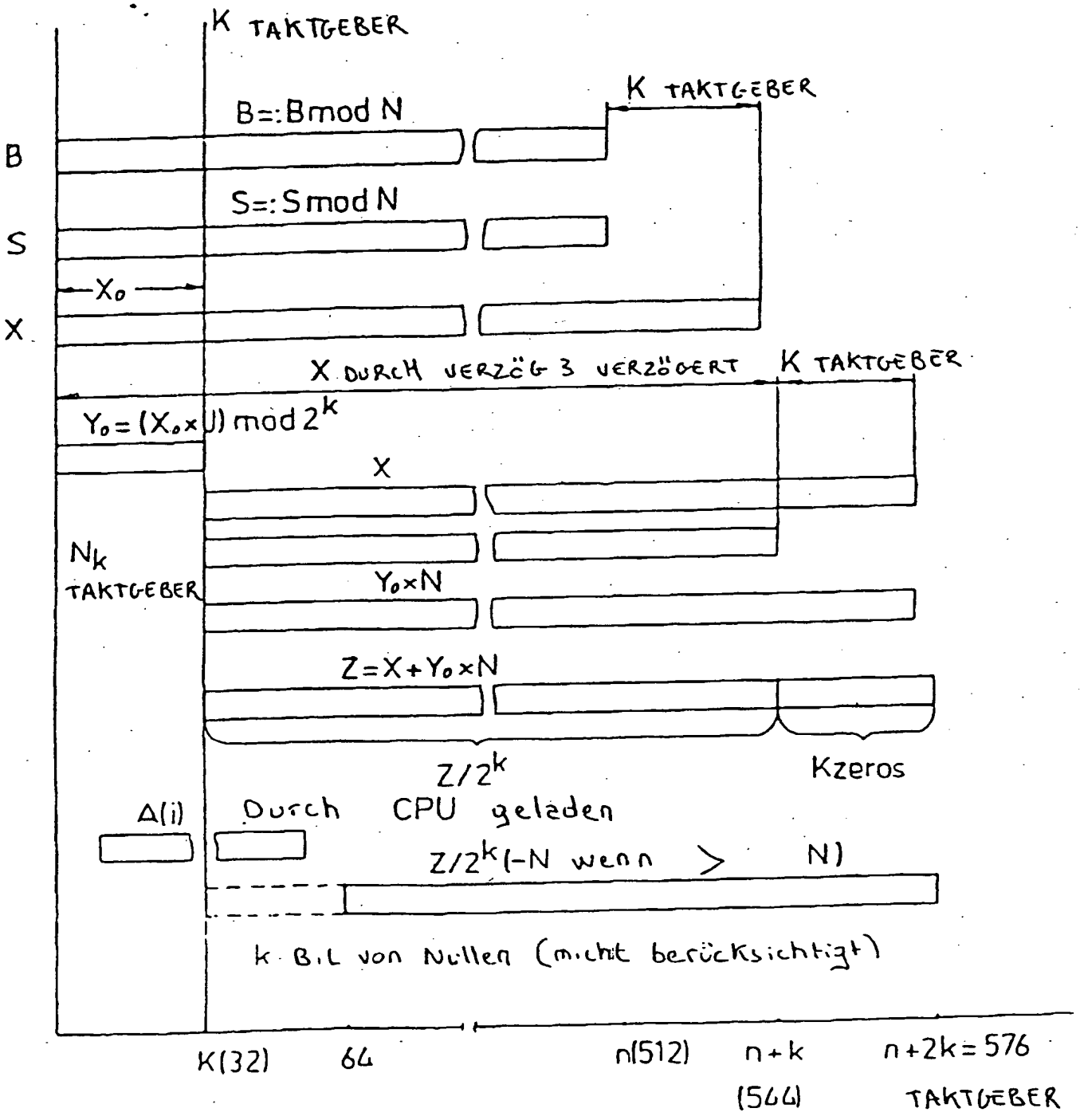
K TAKTGEBER
K TAKTGEBER
B=:Bmod N
B
S=:Smod N
S
X0
X
X DURCH VERZÖG 3 VERZÖGERT
K TAKTGEBER
Y0=(X0×J) mod 2^k
X
Nk
TAKTGEBER
Y0×N
Z=X+Y0×N
Z/2^k
Kzeros
A(i)
Durch CPU geladen
Z/2^k(-N wenn > N)
k B.L von Nullen (nicht berücksichtigt)
K(32)
64
n(512)
n+k
(544)
n+2k=576
TAKTGEBER

Fig. 4

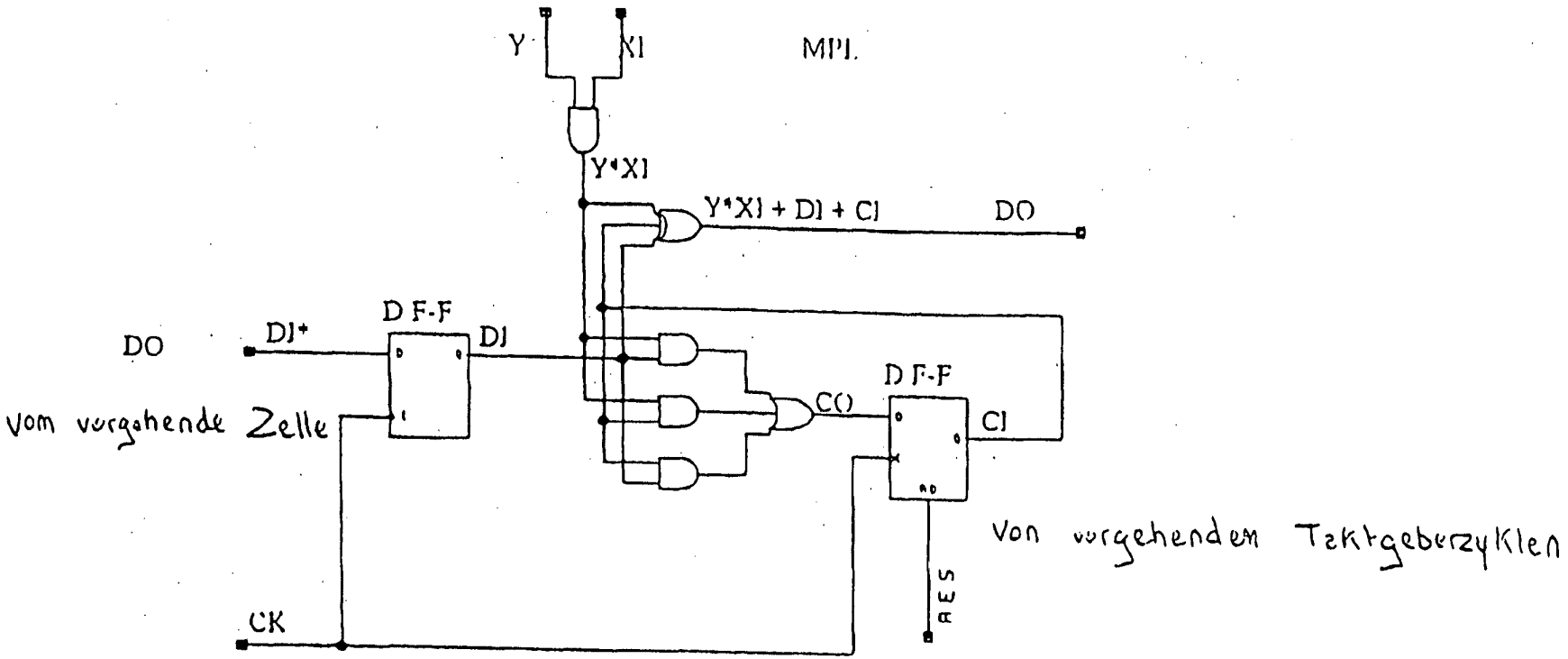
Y
XI
MPI
Y*XI
Y*XI + DI + CI
DO
D F-F
DI*
DO
DI
D F-F
CO
CI
Vom vorgehende Zelle
CK
AES
Von vorgehenden Taktgeberzyklen

Fig. 5

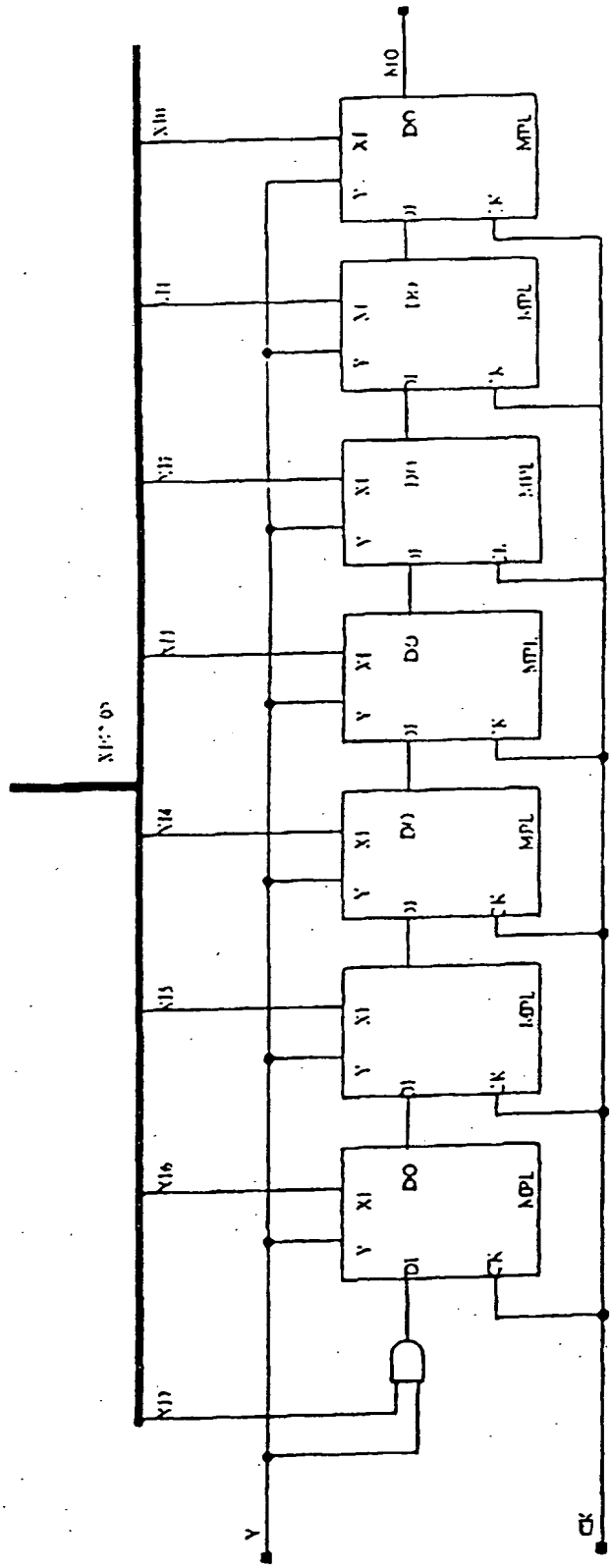

Fig. 6

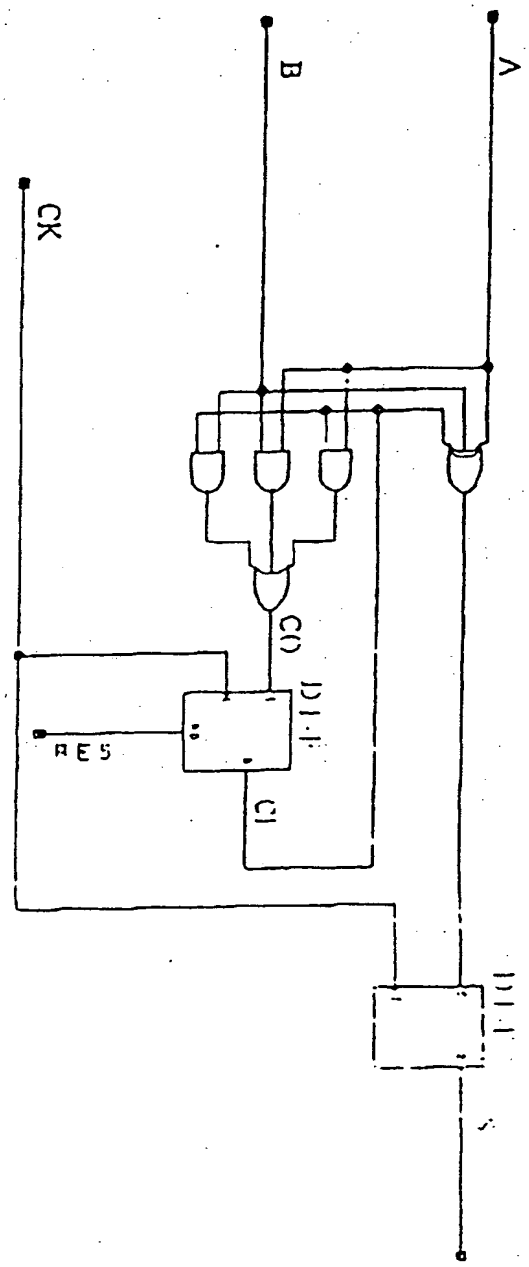

Fig. 7

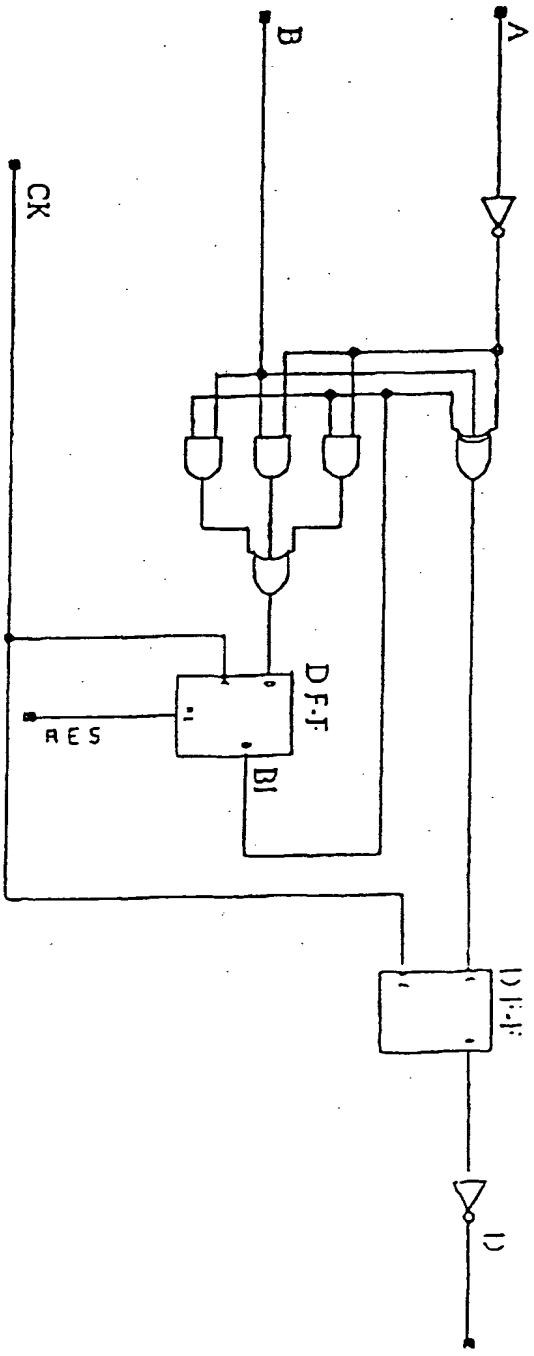

Fig. 8

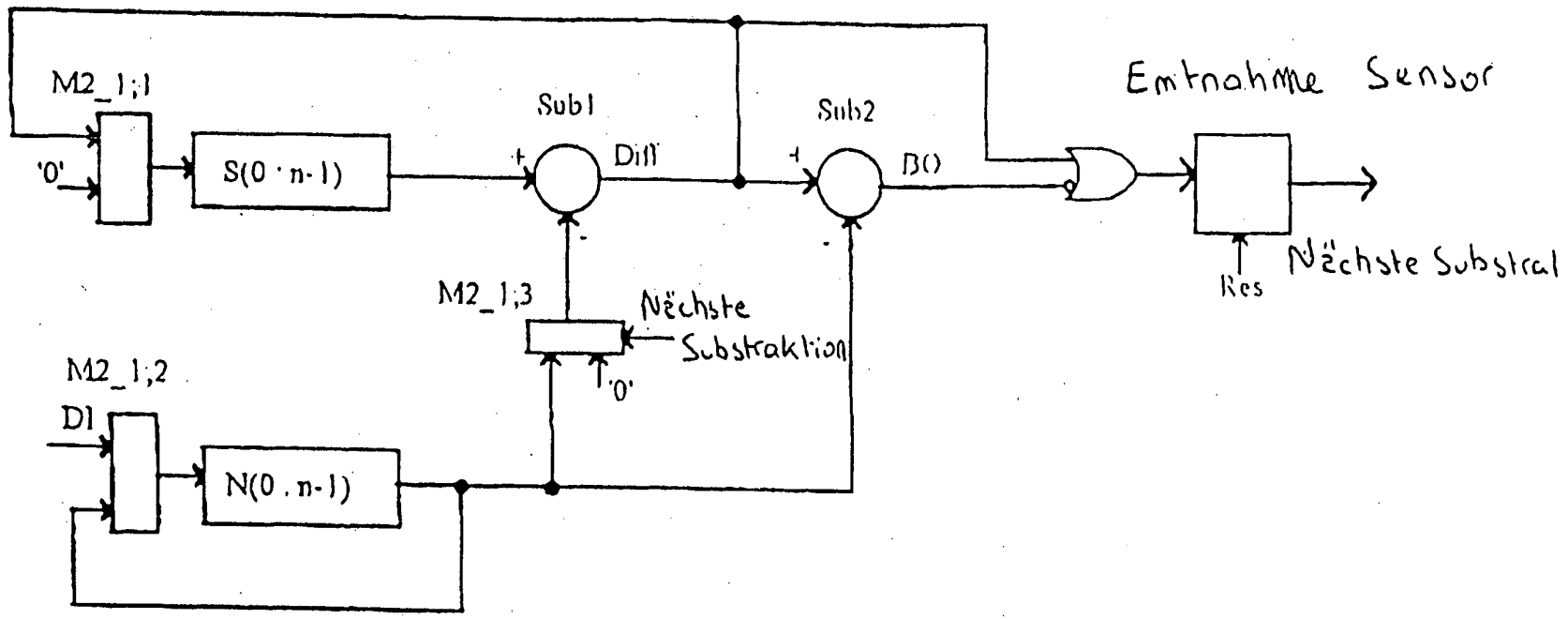
M2_1;1
'0'
S(0 . n-1)
Sub1
Diff
Sub2
B()
Entnahme Sensor
M2_1;3
Nächste
Substraktion
'0'
Res
Nächste Substral
M2_1;2
D1
N(0 . n-1)

Fig. 9